# Table of Contents

**Chapter 1  Introduction**  
What’s in This Book .......................................................................................................................... 2  
About RJS Software Systems ............................................................................................................ 2  
What’s Not in This Book .................................................................................................................... 2  
Document Conventions and Symbols .................................................................................................. 2  
Contacting RJS Software Systems ..................................................................................................... 3  

**Chapter 2  Overview of Enterprise Workflow**  
Architecture and Processing Flow ..................................................................................................... 7  

**Chapter 3  Installation and Configuration**  
System Requirements ....................................................................................................................... 10  
iSeries .................................................................................................................................................. 10  
PC - Server .......................................................................................................................................... 10  
PC - Workstation .................................................................................................................................. 11  
Alternative Enterprise Workflow Configuration .................................................................................... 11  
Web Application Server ..................................................................................................................... 11  
Tomcat 5.x .............................................................................................................................................. 12  
IBM Websphere .................................................................................................................................... 12  
Install Enterprise Workflow Engine .................................................................................................... 12  
iSeries Workflow Engine Installation ................................................................................................... 12  
Configuration Tasks ............................................................................................................................ 20  

**Chapter 4  Workflow Implementation Guidelines**  
Process Analysis .................................................................................................................................... 23  
User Analysis ......................................................................................................................................... 25  
Document Analysis ............................................................................................................................... 26  
Integration with WebDocs .................................................................................................................... 28  
Implementation Plan ............................................................................................................................ 29  

**Chapter 5  Getting Started**  
Login to Enterprise Workflow .............................................................................................................. 32  
Create User Profile ............................................................................................................................. 33  
Define an Access List .......................................................................................................................... 36  
Define Workpiece Class Details .......................................................................................................... 41  
Create a Workbasket ............................................................................................................................ 46  
Design a Workflow Process .................................................................................................................. 48  
Create/Advance Workflow Workpieces ............................................................................................... 56  

© 2006 RJS Software Systems
# Chapter 6 Operations

## Overview

- Find a Workpiece ................................................................. 63
- Create User Group ............................................................... 68

## Process Elements

- Start Point ............................................................................. 73
- Stop Point ............................................................................. 74
- Work Point ............................................................................ 74
- Decision Point ..................................................................... 75
- Exit Point ............................................................................. 75
- Collection Point ................................................................... 75

## Workpiece

- Process .................................................................................. 75
- Priority ................................................................................ 76
- Actions ................................................................................ 76
- Notes ................................................................................... 76
- Attributes ............................................................................ 76
- References .......................................................................... 76

## Login

- .......................................................................................... 76

## Workflow Designer

- Connectors .......................................................................... 77
- Process Menu ....................................................................... 82
- Insert Menu ......................................................................... 84
- Web Service Menu .............................................................. 85

## Workbasket User Interface

- .......................................................................................... 85

## Implementing Workflow Security

- Base Access List .................................................................. 86
- Create a New User .............................................................. 87
- Create a Private User Workbasket ....................................... 89

## Administrative User Interface

- Create Workpiece .............................................................. 90
- Workflow Engine Status .................................................... 92
- Workpiece Status .............................................................. 99

# Chapter 7 Integrating with WebDocs for iSeries

- MAINURL *daara .................................................................. 108
- DOCCHKIN Exit program DOCEXITC ................................ 109

# Chapter 8 Web Service API

- General Processing Actions ................................................. 118
- Workpiece Maintenance .................................................... 118
- Workpiece Lists .................................................................. 126

## Administrative Actions

- Workbasket Maintenance .................................................. 129
- Class Maintenance ............................................................. 131
- User Maintenance .............................................................. 133
- User Group Maintenance ................................................... 134
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Process Maintenance</td>
<td>136</td>
</tr>
<tr>
<td>Lists</td>
<td>138</td>
</tr>
<tr>
<td>Security</td>
<td>141</td>
</tr>
<tr>
<td>User Authentication</td>
<td>141</td>
</tr>
<tr>
<td>User Authority</td>
<td>141</td>
</tr>
<tr>
<td>Access List Maintenance</td>
<td>143</td>
</tr>
<tr>
<td>Privileges</td>
<td>146</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction
Introduction

This chapter provides basic information about what is in this manual, including an overview of RJS Software Systems, document conventions and how to contact RJS.

What's in This Book

This reference manual provides detailed information on how to install, configure and administer Enterprise Workflow. Here is what's included in this book.

- Chapter 1 - Introduction: describes this reference manual and how to use it effectively.
- Chapter 2 - Overview of Enterprise Workflow: describes Enterprise Workflow, the benefits of using Enterprise Workflow and licensing of the product.
- Chapter 3 - Installation and Setup: provides step-by-step instructions to install and configure Enterprise Workflow and required supporting tools, and, if necessary, how to uninstall Enterprise Workflow.
- Chapter 4 - Workflow Implementation Guide: provides guidelines for analyzing your workflow processes, associated users and reference material.
- Chapter 5 - Getting Started: provides a quick start to using Enterprise Workflow for the first time. Detailed instructions are found later in this manual.
- Chapter 6 - Operations: provides detailed instructions on administering Enterprise Workflow processes, users, access lists, workbaskets, and security.
- Chapter 7 - Web Service API: describes the various methods available to develop your own customized interfaces and extensions to Enterprise Workflow.

About RJS Software Systems

RJS Software Systems is a privately-held software and hardware company dedicated to providing high-quality System i/i5/iSeries, Client/Server and web-based products and customer services. Customer Service is central to the company’s objective. Read more about us on our web site, RJS web site.

What's Not in This Book

This reference manual does not describe the following:

- Configuration and use of individual document input equipment (i.e., fax or scanner). Refer to manufacturer instructions for help.
- Use of individual web browser(s).
- Explanation or help with OS/400 commands, outside the scope of need as it pertains to WebDocs.

Document Conventions and Symbols

The following document conventions are used throughout this reference manual:

- Acceptable hardware server for Enterprise Workflow could be noted as AS/400, iSeries, i5 or System i. For simplicity, all references to the server in this manual will default to the term iSeries unless the notation is specific to a specific server model.
- With System i hardware, currently 5XX models, the IBM has given the operating system a new name of i5/OS. On iSeries hardware the operating system is known as OS/400. For simplicity, all references to the operating system in this manual will default to OS/400 unless the notation is specific to the new i5/OS.
- Titles of documents appear in Italics. Italic type is also used to indicate information that varies...
by circumstance.

- The document version, shown on the first page of this manual, denotes both the Enterprise Workflow software version this manual represents and document draft for the release. Document version will be in the format of X. YY.N. Where X. YY represents the Enterprise Workflow software version and N represents the document draft number. For example, if the document version is noted as 1.62.2 then the information in this manual is current as of Enterprise Workflow version 1.62 and is the second release of the manual for software version 1.62.

**Contacting RJS Software Systems**

For technical support, please review the following information before contacting RJS Software Systems.

**For Technical Support with WebDocs**

Because of the complexity of dealing with the various iSeries connectivity environments, please gather and organize as much information as possible on the problem prior to contacting RJS Software Systems for support.

If you have a question about an WebDocs operation, first browse through the Online Help to try to find your answer prior to calling RJS Software Systems.

**Contact Information**

Telephone support is available on normal business days from 8:00 am to 5:00 pm central time.

- **952.898.3038** Voice
- **952.898.1781** Fax

Support is available via email at [support@rjssoftware.com](mailto:support@rjssoftware.com).

You may also find the answer to your question on our web site: [RJS web site](http://www.rjssoftware.com).
Chapter 2

Overview of Enterprise Workflow
Overview of Enterprise Workflow

Enterprise Workflow is intended to increase productivity and accuracy of virtually any business process. You can think of it as a task-master, constantly moving a document or item through the process.

Enterprise Workflow can be used to manage many different business processes within your organization. For example, Enterprise Workflow could be used to manage the purchasing within your organization. Typically purchasing consists of: creation of purchase orders, approval of the purchase orders, verification of the receipt of the purchased items and payment of the invoice. Although this process seems straight-forward, each individual step may contain additional controls or require multiple people to sign-off on the document(s). For example, most organizations require multiple signatures or approvals for larger purchases. Additionally, some organizations require additional steps within the purchasing process, such as quality control to ensure the quality of the items meets required manufacturing specifications. Purchasing is just one of many types of processes that can be controlled by Enterprise Workflow.

Below is a sample of a workflow for the claims processing process for an insurance company. Again, this is just one of many possible processes that could be managed with Enterprise Workflow. Another way to recognize a process that can be managed by Enterprise Workflow is to consider those documents or work items that are moved from one employee to the next using a routing slip or routing envelope.
**Sample Workflow for Insurance Claims Processing**

Enterprise Workflow controls business processes through a combination of process definitions, process users and supplemental programs. You will define, in detail, each step (or work point) of the process, what is required to complete each work point, the order of the steps, decisions to be made within the process, and how the process is complete.

Workflow processes are not necessarily linear in nature and may have multiple exit points. Enterprise Workflow can be used to manage one or many processes within your organization. Each process is designed using our Workflow Designer interface - a graphical tool that allows you to detail any process.

Enterprise Workflow is designed to provide you with business process management and automation. It is designed to be easily integrated with your current applications and can be customized to guarantee Enterprise Workflow fits within your process rather than forcing you to fit your process to Enterprise Workflow!
Enterprise Workflow is a complete Java framework which can be used for easy integration with existing front-end applications, or rapid development of advanced browser-based user interfaces for business process management and task fulfillment. Rapid development of customized user interfaces is accomplished through the use of an extensive set of APIs included with Enterprise Workflow.

**Architecture and Processing Flow**

How does Enterprise Workflow provide you with a method to automate and document your key business processes? First, it uses a web service-based Application Programming Interface (API) where all actions, events and status queries are driven through a single, web service application. It includes a configurable (by the customer), browser-based user interface that is a set of Java Server Pages (JSPs) that access the Enterprise Workflow APIs.

Initially, Workflow Designer is used to document the details of a process; including all required “action” steps, decisions and processing exit points. The details of your process design are stored in a database shown as “Steps” in the architecture shown in the Enterprise Workflow Architecture graphic, shown below. Additionally, you will define the metadata associated with each class of workpiece; a workpiece represents an item, such as a manufacturing order or insurance claim, which will be advanced through the process. Metadata includes key “Values” or attributes that identify the workpiece and external document references that are linked to the workpiece.

Yet, how does workflow use the combination of workpiece classes and workflow steps to manage your business process? From a browser, any authorized user can create a new workpiece for a
process. As you can see on the architecture graphic, that request will be filtered through the HTTP server to your web service (Tomcat or Websphere) and then to the Enterprise workflow application.

Enterprise Workflow initiates the processing based on the workflow process design.

The advancement of a workpiece, shown in the Workpiece Processing graphic, is completed by determining the current step for the workpiece, the action either selected by a human or programmatically and the next step in the process. If the next step is a decision point, then the workpiece advancement will be determined by comparing the values linked to the workpiece with the comparisons established in the decision point. If the advancement is done programmatically, Enterprise Workflow temporarily halts internal processing and calls an external program (an exit point) which allows you integration with other applications and data.

Customers can customize Enterprise Workflow through the provided APIs and the creation of external programs that are called from exit points in the process. Enterprise Workflow includes a full set of APIs that can be called from any application that can build an XML and send it over HTTP protocol. On the server, Exit Points, as noted in the process design, call supplemental programs that perform tasks outside the realm of the Enterprise Workflow application and interfaces.

One last note, Enterprise Workflow constantly records all activity, from workpiece creation to the final advancement to end step in the history logs. This enables you to audit any and all Enterprise Workflow Activity.
Installation and Configuration

This chapter is dedicated to the basic installation and configuration necessary to be able to use Enterprise Workflow. All persons that will be responsible for administering or installing Enterprise Workflow software should review this chapter. Although this chapter does not cover the installation prerequisite software - Tomcat application server, Java Virtual Machine (JVM) and a SQL database, it does provide direction as to where and how these packages can be obtained.

The normal installation process is as follows:

1. Review software requirements
2. Install any missing prerequisite items
3. Install Enterprise Workflow server engine
4. Configure Enterprise Workflow
5. Install Workflow Designer on all workstations that will be used to design workflow processes

System Requirements

Enterprise Workflow has been developed to be platform independent. Currently we have certified two server platforms for Enterprise Workflow: IBM iSeries systems or Windows XP. We have listed the software/hardware requirements for both environments. You need only to review the requirements for the server environment you intend to use for Enterprise Workflow.

In addition to the server, you will have workstations used by your workflow users. If you intend to use the Enterprise Workflow browser environment, you will find the requirements quite basic. If you intend to develop customized applications to tightly integrate Enterprise Workflow with your current application environment, your workstation requirements may be different than those we have listed.

iSeries

If you intend to use the iSeries as your Enterprise Workflow server you will need the following environment:

- Tomcat 5.x or IBM WebSphere web services environment installed and configured
- OS/400 V5R1 or higher

Note: RJS Workflow is an AXIS application. The Enterprise Workflow software class libraries come complete with AXIS version 1.0. Your choice of Web Application Server should be compatible with AXIS 1.0.

PC - Server

If you intend to use a Windows system as your Enterprise Workflow server you will need the following environment:

- Windows XP, 2000 or 2003
- Java Virtual Machine (JVM)
• Tomcat 5.x web services environment installed and configured
• Java runtime 1.4.2 or higher installed
• Accessible Database: MySQL, SQLServer, or Microsoft Database Engine (MSDE)

PC - Workstation
For those workstations in which you intend to use Enterprise Workflow Designer tool you will need the following:

• .NET 1.1 runtime files; **note:** at this time 2.0 does not work
• TCP/IP connection to Workflow engine

For each workstation that will be used for general access to the Enterprise Workflow server you will need the following:

• Internet Browser with an active internet connection

Alternative Enterprise Workflow Configuration
Enterprise Workflow requires the presence of a web services application (i.e., Tomcat) but your web services application does not have to be installed on the same server as Enterprise Workflow. It is possible for you to run Tomcat from a Windows server and Enterprise Workflow from an iSeries server. With this configuration Enterprise Workflow will use the iSeries database (known as DB2 Universal Database or DB/400).

Web Application Server
RJS Enterprise Workflow is a web application. It must run under the cover of a Web Application Server. Apache Tomcat and IBM Websphere Application Server are two examples of such web application servers.

The workflow engine is a web service application that works with Apache AXIS to provide SOAP messages. SOAP is an object access protocol that trades program call information through HTTP or other common protocols. In this instance HTTP is the transport protocol. The web service portion of RJS Enterprise Server operates under a web application context name "WorkflowWebServices". The AXIS application works within this context in a path named "services/WorkflowWebService".

Web Services provide their schema, a description of their interface protocol, through an XML file. This XML file conforms to another schema called a WSDL file. To get the WSDL file of the RJS Enterprise Workflow engine web service use a URL something like this:

    http://hostname:port/WorkflowWebServices/services/WorkflowWebservice?wsdl

    where hostname and port match the configuration of your web application server.

RJS Enterprise Workflow also includes a web application for accessing the workflow engine through some example JSP pages. This web application works in the application context named "WorkflowWebServiceClients". The JSP pages are in a path within this context named "WorkflowWebServiceClients". The two "launch" pages from which all the others are launched are the "Admin.jsp" and "Workflow.jsp". Copy the following two URLs to your web browser and substitute the host DNS name or IP address and port for your web application server.
configuration:

http://hostname:port/WorkflowWebServiceClients/WorkflowWebserviceClients/Admin.jsp

http://hostname:port/WorkflowWebServiceClients/WorkflowWebserviceClients/Workflow.jsp

Later sections of this document specify how to set up the contexts of your web application server to support the RJS Enterprise Workflow engine web service application and JSP application.

**Tomcat 5.x**

If you have not yet installed the Tomcat web application server you can download Tomcat from the Apache Software Foundation website. Follow the instructions there to load it on your system.

http://jakarta.apache.org/site/downloads/downloads_tomcat-5.cgi

Configure the "server" address and port. This will be the address used by your server. Remember these values as you will need them during the installation of Enterprise Workflow.

**IBM Websphere**

If you intend to use IBM's Websphere Application Server refer to the IBM documentation for help installing and configuring Websphere.

**Install Enterprise Workflow Engine**

**iSeries Workflow Engine Installation**

The workflow engine is a web application that runs under Apache Tomcat Server or Websphere Application Server. It also requires a database with a JDBC connector.

RJS Workflow has been tested with Tomcat 5.0.28 and Java VM 1.4. Download and install these environments onto your target machine. It should also work with WAS 5 or better and JVM 1.4.

RJS Workflow has been tested with MySQL, SQL Server and DB2UDB (also known as DB/400) for iSeries.

**iSeries Installation Note:** If you will be installing Enterprise Workflow on an iSeries server you must have a mapped NFS drive (Netserver share) configured and accessible at the time of the installation.

Use the following instructions to complete the Enterprise Workflow installation.

1. Download the Enterprise Workflow installation program from the RJS Software Systems website using the following URL:

   http://www.rjssoftware.com/files/rjsflow/workflowsetup.exe

   You will be presented with the welcome screen.
2. Click on the **Next** button to begin the installation process.

You will be prompted to select the desired components to install. The Workflow server should be installed on one server, either iSeries or Windows. You should not install the Workflow Server on all workstations. The Workflow Designer should be installed on those workstations that will be used to design your workflow processes.

3. Choose the **desired components** and then click **Next** to continue the installation process.

You will be prompted for the location of the Tomcat web applications directory. Typically, the applications directory is a sub-folder of the main Tomcat directory. If you are installing on an iSeries system this directory will be found on your mapped NFS drive.
4. Click **Next** to continue the installation process.

You will be prompted to select or type Start Menu Folder to hold the program shortcuts.

5. Click **Next** to continue the installation process.

You will be prompted to enter the Enterprise Workflow license key. This key is usually sent to you via email, and due to its length, is best copied from the email message to this dialog. Copying the key will eliminate typing errors.
6. Copy the Enterprise Workflow License Key into License Key edit box and then click Next to continue the installation process.

You will be prompted to define your SQL database configuration.

If you are installing the Enterprise Workflow server to a Windows server you should select either Microsoft SQL Server or MySQL as your server type. The SQL Server option is appropriate if you have either SQL Server database or just the Microsoft Data Engine (MSDE) configured on your Windows server. MySQL is an open source database that you can use to manage your Enterprise Workflow. If you are installing onto an iSeries server, select the iSeries option which will use the universal database included in the iSeries operating system.

For the Server IP enter either a valid DNS name or IP address where the database is located.

For SQL Server and MySQL databases you need to enter an IP port number. The default port numbers for the two databases are shown on the dialog. Your database administrator can tell you the actual port number to use for your environment.

7. Select the appropriate Server type, enter the database IP Address and, if necessary, enter the database Port Number. Once completed, click Next to continue the installation process.
Next you will be prompted to enter two sets of user IDs and passwords for database access. The first set will be used immediately to create necessary Enterprise Workflow database tables and records. The second set will be used by Enterprise Workflow after the initial installation phase. For SQL Server the default Administrators Username is 'sa' and the password is blank. For MySQL the default Administrators Username is 'root' with a password of 'admin’. Although the iSeries is shipped with default user profiles, most shops will create individual user profiles during the initial setup of the server. You will need to speak to your network administrator to confirm the actual Administrator Name/Password that should be used for your SQL database.

The second set of username/password can either be the same profile as used for the installation or you can create a new profile to be used exclusively by Enterprise Workflow. This decision will be between you and your network administrator. If you decide to use a new user profile for the day to day operations of Enterprise Workflow then you will need to ensure the username and password entered here are used to create a new profile in your database environment before you first log into Enterprise Workflow.

8. Enter Administrator Username/Password for your SQL database to be used to create the necessary SQL tables and enter a Workflow Username/Password that Enterprise Workflow can use on an ongoing basis to create/modify SQL database records. Then click Next to continue the installation process.

The Enterprise Workflow installation program uses ODBC connections to access the selected SQL database/server. If you do not have the required ODBC connection configured on your PC for the installation program to use then it will create it’s own. If you receive the following prompt you can answer Yes if you believe there is a useable ODBC data source.

Note: the next four steps, 9 through 12, will be skipped by the installation program if it can find an acceptable ODBC data source configured on your system. In this example we see the system will be attempting to create a MySQL ODBC data source.

9. Click Yes to select an existing ODBC data source to use during installation. If you click Yes
skip to step 13 of the installation process. Otherwise, click No to set up a new ODBC data source for your SQL database.

You will be presented with a Welcome screen for the ODBC setup wizard.

10. Click Next to continue the ODBC setup.

You will be prompted to select the type of setup you prefer. In general, choosing the Typical option will allow you to create a valid ODBC data source for the Enterprise Workflow setup process. If you or your network administrator have specific ODBC connection requirements, you can choose either the Complete or Custom options. This installation only covers the Typical ODBC setup process. Refer to your ODBC data source help for details on the more advanced data source options.

11. Select Typical as the type of setup then click Next to continue the ODBC data source setup.

You will be prompted to begin the ODBC data source installation process.
12. Click **Install** to install the ODBC data source.

   You will be prompted when the connection wizard has completed the installation of the ODBC data source.

13. Click **Finish** to complete the ODBC data source installation process.

   You will be asked to enter the details of your Tomcat installation.

   First, is the URL for the Tomcat Web Service which is the web application that will service Enterprise Workflow. A sample URL is shown below:

   [http://10.1.1.6/WorkflowWebServices/services/WorkflowWebservice](http://10.1.1.6/WorkflowWebServices/services/WorkflowWebservice)

   When Tomcat is installed, it is assigned an IP port number. Your network administrator should be able to tell you the port number used for your Tomcat installation. The default port number is 8080.

   If you intend to use email for workflow notification purposes, you will need to enter the details of your SMTP Mail Host.
14. Enter your **Tomcat Webservice URL, Port Number and your SMTP Mail Host address** then click **Install** to install Enterprise Workflow.

Once installation is complete you will be presented with a success message.

15. Click **Next** to complete the installation process.

You will be presented with the final completion message.
Your Enterprise Workflow is now installed. Remember, if you entered a new SQL database/server Username and Password for Enterprise Workflow, you must now create that user profile before you attempt to log onto Enterprise Workflow. Refer to your database reference material for help creating the new profile.

**Configuration Tasks**

During the installation process of Enterprise Workflow some default objects are created for your initial use. The following items are created for you:

- **An Administrator user profile** is created which has full authorization to all privileges and objects. The user name is *Administrator* and the password is *Administrator*. Both user name and password are case sensitive.

- **An Admin access list** is created which can be used to assign administrative users with all Enterprise Workflow privileges.

- **Several sample workbaskets** are created: *Approval 1*, *Approval 2*, *Approval 3*, *Approval 4*. These workbaskets can be used to set up test or sample workflow processes.

- **An ApprovalClass workpiece class** is created. Workpiece classes are used to define the types or categories of items that will be moving through Enterprise Workflow.

- **An Admin users group** is created which can be used to manage all of your users that will be given administrative privileges.

Enterprise Workflow is operational immediately following the installation process. However, you should complete the following tasks to customize your Enterprise Workflow for your business:

- Create an implementation plan (this can be done by an RJS site survey). Once completed, you can then complete the remaining items on this list. Refer to the *Workflow Implementation Guidelines* chapter for help.

- Define your security plan and create the necessary access lists.

- Create all user profiles. Define workpiece classes that will be used for your initial workflow processes.
Define workbaskets that will be used for your initial workflow processes.

Define your initial workflow processes using Workflow Designer.

Refer to both the Getting Started and Operations chapters for help in configuring and using Enterprise Workflow.
Chapter 4

Workflow Implementation Guidelines
Workflow Implementation Guidelines

There are several areas of consideration as you begin using Enterprise Workflow: Document Analysis, Process Analysis, User Analysis, Integration with RJS WebDocs document management system and an Implementation plan. The sections below address each of these areas and provide suggestions on the type of information you will need to gather from your users as well as helpful guides and forms to be used during the planning phase.

Well laid plans will make a difference in the ease with which you implement Enterprise Workflow. We recommend that each workflow process should be separated into four phases: plan, design, test and rollout.

This section is intended to help you and/or a consultant properly plan for a successful Enterprise Workflow implementation. The bulk of the information is dedicated to helping you intimately understand and document the requirements of your process(s). All of the information you gather will be used to more thoroughly design your workflow process and to define your implementation plan. If a consultant will be working with you, the consultant will work with you to complete the site survey.

Process Analysis

Although you can use Enterprise Workflow to control almost any business process, you may want to focus on one or two processes as your flagship workflow processes. Or, you may choose to implement a simple workflow in a non-critical process in your company to gain experience in implementing workflows. This will reduce the scope of your projects while you learn the intricacies of Enterprise Workflow. Create a list of the top processes you are considering for workflow and then look at the priorities, potential cost savings and staff availability to help you determine what processes should be tackled in the initial implementation period.

Once you decide on the processes that will be controlled through Enterprise Workflow, determine how they relate to the day to day tasks within the business. Make sure you understand how to integrate the new processes with existing processes.

In the plan, document how the process works today and how that will compare to the new automated workflow process. If there will be changes to the process either to streamline the process or because of the new abilities available through Enterprise Workflow, make sure and document those differences. Be sure to include the amount of time it typically takes using the current process to complete the task and what your expectations are to complete the process using Enterprise Workflow. Remember, you may have more than time savings by using Enterprise Workflow. Other potential benefits of using Enterprise Workflow include improved accuracy, cost savings, client/internal security, improved business communications, etc.

Diagram the process with as much detail as possible. We recommend using some method or form of flow charting to diagram the process. The diagram should include the various paths, decisions to be made along the way, possible exit and end points, and the users, or groups, associated with the workflow process.
You may want to include a written description of the process, either in narrative form or at least detailed descriptions of each step in the flow chart. As you look at the process steps consider if there is any additional electronic (or automatic) processing that will be necessary. For example, when processing accounts payable invoices a percentage of invoices can be automatically approved simply by a combination of locating a pre-approved purchase order and confirming the invoice amount is less than X dollars. If this is the case, a program may be able to automatically complete the payment of the invoice and move the invoice to the end of the process without any additional work for the AP clerk. If there are certain points in the process where you would like to use external programs, then those instances should be included as an exit-point in the process.
User Analysis

Regardless of the process, there will be from one to $N$ users involved in the process. You will need to determine which users, or groups, will be part of the process. Also, consider if there are other personnel that will need access to the process for either observation or process control. For example, your executives may need access to a financial audit practice but will not be intimately involved in the process. In that case, you will probably grant most executives the ability to view the process and associated documents but not grant them the authority to make changes. By giving them the rights to view but not change you can ensure that the experts of a particular process, in this case the financial audit, are the users that actually complete the process.

Are the users of the workflow process all part of a common group, for example Claims Processing? If so, you can easily define your authorities to various tasks and documents through groups. Also, if the process is one where you may have many users that can all process one or more steps in the process than these users can be grouped and accesses defined for their group. If, on the other hand, you are defining a process where the users are not all part of a common group you may need to define your access rights according to individual users.

As you look at the users of the process you should also consider the security of the process and documents by users and groups of users. Look at the security of the entire process, individual steps and documents. Enterprise Workflow allows you to define security at a granular level – for example, whether a user can create a new work piece or view a work piece. You should also consider if there will be users of the process that are outside of your organization such as your customers. The main document security is defined through our WebDocs document management system but security can also be accomplished by securing either the workflow process or steps associated with the documents.

To ensure a complete picture of the workflow process users and supervisors of the process should be interviewed. The plan should document all tasks, authorization requirements and decisions that occur in the process by user/groups. If possible, categorize based on user groups or roles which will be useful when actually defining process users.
Document Analysis

Invariably, each workflow process will have certain documents associated with the process. Your plan should document the types of documents that are an integral part of the process as well as any ancillary documents that may be needed.

Document sources are varied. Sources can be electronic or paper, standardized or free-form, new or existing documents. Those documents that are currently in paper form will most likely need to be digitized – converted to a digital image. Usually this is accomplished through the use of a scanner.

Scanning documents brings up several other items to consider. Are the documents multiple pages? Will they be scanned individually or in bulk? How will they processed and categorized after they are scanned? Can you use optical character recognition (OCR) to read key pieces of information off of the form such as name, ID number or phone number? To take advantage of OCR it is best if the information is presented in a standard form; for example, an insurance form.

Consider each document in the process and determine if you will only be including new documents or will you back-fill with existing documents. If you choose to back-fill, you need to determine how far back in time you will be pulling documents. It is important that everyone in the organization understands which documents will be used in the new process and which will remain in the previous locations or system, so document it clearly.
In your plan, explain how each document is used. The more information you provide, the better your overall process will be defined.

Try to quantify the documents that are associated with each workflow process. On an average day, how many documents will go through the system? Determine the maximum number of documents you expect in the process on a daily basis? Weekly? Monthly? What are the consequences if you are unable to process the documents? Ascertain who will be responsible for the processing of the documents. Outline whether one or multiple people will be responsible for processing and handling of the document(s) for the workflow. Is one person responsible for the document from beginning to end? Or, are multiple people required to move the document through the workflow process?

Determine if any of the documents contain sensitive information. If some or all of the documents must be restricted to certain users or groups of users, your plan should include necessary precautions to protect the document information.

Your plan should detail document source, timeliness, security concerns, how the documents are used and the users of the documents. Use the following table to help record the types of business documents that will be included in this process.
**Process:** Purchase Orders

<table>
<thead>
<tr>
<th>Document Type</th>
<th>Source</th>
<th>Description</th>
<th>User(s)</th>
<th>Security Requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>Supply Requisition</td>
<td>Electronic Form</td>
<td>Form completed by employee to request purchase of equipment or supplies.</td>
<td>All</td>
<td>None</td>
</tr>
<tr>
<td>Purchase Order</td>
<td>Programmatically created</td>
<td>Created once first round of approval has completed</td>
<td>Purchasing Dept.</td>
<td>Specific Users only</td>
</tr>
</tbody>
</table>

**Integration with WebDocs**

Although Enterprise Workflow is application-neutral, RJS strongly recommends integrating Enterprise Workflow with a document management system such as WebDocs. If you will be using a document management system then your plan needs to detail how the software will be integrated with Enterprise Workflow.

If you will be integrating with WebDocs, then you need to understand if all workflow users will also be WebDocs users and vice versa. You may find that your workflow users are a subset of the WebDocs users. By defining the same User IDs and passwords in Enterprise Workflow and WebDocs you can eliminate the need for users to log into both applications. Plus, it will reduce the user maintenance required by your IT staff.

Refer to the WebDocs manual for detailed descriptions on how to configure and setup WebDocs for use. Your workflow plan should include requirements for the different types of documents that will be stored, what index fields (document lookup keys) will be necessary for each type of document and the directory or folder structure needed when checking supporting workflow
documents into WebDocs.

**Implementation Plan**

The implementation plan should include not only the low-level details of process, users, etc, but also high-level details such as who will be responsible for the overall project and the schedule of events. The schedule should include time to plan and interview, design, test and then rollout the process. Also, the schedule should note any time constraints or concerns. For example, does the rollout of the process have a drop-dead date that must be met? Consider the repercussions if important dates are missed and include alternative plans when there are absolute milestones to be met.

Select a group of people that will be closely involved in the planning, design and testing of the workflow process. Make sure that you include person(s) for all key areas of the process so that all factions are well represented. This will ensure a more thorough design and test of the process.

Specify how you intend to handle the transition. Use the planning stage to help you anticipate any transition issues you may encounter and then create a transition plan that will address these issues. Common transition issues include communicating old versus new process steps and defining how historical business information will be included. The more you can communicate either verbally or through well-documented plans will help eliminate any confusion among employees. Decide what will be done with historical business documents: leave them as they are or integrate into the new systems. Integration may mean you will need to convert back-documents to digital format through scanning of the documents and then checking them into the document management system. It is imperative that you define a line of demarcation with regard to how/where users will find historical business documents. Having a clear line in the sand will help users determine if a customer’s invoice is in the filing cabinets or stored on-line.

Your rollout plan will go more smoothly if you include training in the overall rollout of the workflow process. Determine who will train your users, where the training will be conducted and if there will be any type of follow-up training for your users. All of this training should be included in the project timeline.
Chapter 5

Getting Started
Getting Started

This section has been designed to walk you through all of the common tasks associated with Enterprise Workflow. Some of these are administrative and some are focused on the daily tasks your end-users will complete in Enterprise Workflow. In addition, the first seven tasks are designed to be completed chronologically and will guide you through the basic steps necessary to configure and use Enterprise Workflow. The remaining two exercises are supplemental tasks that are useful, but optional.

By completing the first seven Getting Started exercises you will have a solid understanding of the configuration tasks necessary to setup your Enterprise Workflow environment and how to insert/advance work items within a defined workflow process. The graphic below represents the seven primary exercises in the Getting Started section. At the beginning of each of the first seven tasks, you will see this graphic with an arrow showing you which step you will be reviewing next.

Seven Basic Enterprise Workflow Getting Started Tasks

By learning the basic steps described in the first seven Getting Started tasks you will gain a comprehension of how to administer and use Enterprise Workflow. Obviously, these sections will not teach you all of the details associated with each of the tasks, but you will have a general understanding.

The eighth task Find a Workpiece will be helpful to any Enterprise Workflow user. It will teach users how to locate a workpiece and check on the status or work associated with the item.

The final task, Create User Group is an administrative task that you may or may not use within Enterprise Workflow. User groups can simplify the necessary steps to securing workflow assets.
Login to Enterprise Workflow

Before you are able to do any workflow processing you must first log into the Enterprise Workflow server. Follow the steps below:

1. Open your browser and enter the URL to access the Enterprise Workflow server. Use the IP address and port number that was used during the configuration of Enterprise Workflow. A sample URL is shown below.

   http://10.1.1.10:8089/WorkflowWebServiceClients/WorkflowWebServiceClients/Admin.jsp

   The above URL will take you to the default administrative page of Enterprise Workflow. Replace "Admin.jsp" with "Workflow.jsp" to access the end-user interface of Enterprise Workflow.

   You will be presented with a screen similar to the one shown below.

2. Enter your user ID and Password.

   If you do not yet have an ID and password, Enterprise Workflow configures a default user ID Administrator during the database setup. The password is "Administrator".
Once logged in, your screen will look similar to the one shown below.

![Workflow Administration](image)

**Successful User Login**

Congratulations, you have logged into Enterprise Workflow.

**Create User Profile**

Similar to many network servers or software servers, each user must have a unique profile defined to use Enterprise Workflow. You can use a combination of users and user access lists to create a secure environment that meets your security requirements for business processes and documents. In this section you will learn how to create a new user account for Enterprise Workflow. For detailed help on the intricacies of user accounts refer to the administration section of this document.

1. Log into the Enterprise Workflow Admin page using a user profile that has administrative authorities.

   During the installation process of Enterprise Workflow the user Administrator is created and given full rights to all Enterprise Workflow functions. The default password for Administrator is Administrator.

2. Click **Maintain User Account** from the list of actions on the left.
You will be presented with a screen similar to the one below.

3. Select **New** from the User Name drop-down list.

You will be presented with a screen similar to the one shown below.

4. Enter a **User Name** - a user ID, for the new user.
The value you enter into User Name will be used as the User ID by the user each time they log into Enterprise Workflow. This value is case sensitive which means if you use proper case for the user name in the description field then the user will need to use proper case when logging onto Enterprise Workflow.

5. Enter **Description** for the new user.

The Description is often used to provide the full name of the user that will be using the profile.

6. If you have already defined access lists for your users, select the appropriate access list that provides the correct level of security for the user. Otherwise, accept the **default** access list.

7. Enter a **Password** for the new user.

Passwords are case sensitive. When logging onto Enterprise Workflow users will be required to enter the password exactly as you have entered it here. Your completed User Account will look similar to the one shown below.

8. Click **Add** to save the new user account.

Once the new user profile is saved you will be presented with a screen showing the profile, the user that created the new profile, and a date/time stamp that reflects the creation time.
Congratulations, you have created a user profile for Enterprise Workflow.

**Define an Access List**

Access lists are used to provide security to your various processes and functions within workflow. In this example we will define a basic access list that could be used by the Accounts Payable personnel to process purchase orders. Refer to the administrative section of the manual for details regarding access lists.

Below are the instructions to defining an access list.

1. **Login** to the Admin page of Enterprise Workflow.
   
   Refer to the [Login to Enterprise Workflow](#) instructions for help with this.

2. Click **Maintain Access List** from the list of actions on the left.
   
   You will be presented with a screen similar to the one below.
3. Select **New** from the Access List Name list box.

You will be presented with the following screen.

4. Enter an **Access List Name**, such as *AP*.

The value or name that you enter into Access List Name will be the access list name that will you see in all drop-down selection boxes throughout the product.
5. Enter a **Description** for the access list, such as *Accounts Payable*.

Access lists are used to control security to the workbasket and the items contained within it. Your screen will look similar to the one shown below.

![New Access List](image1)

6. Click **Add**.

![Add Button](image2)

The new access list will be created. You will then be presented with a screen that allows you to define which users groups and privileges are allowed for the new access list. You may assign individual privileges either en mass using a user group or individually for a user. Usually it is easier and reduces maintenance of your security if you define privileges by user groups rather than individual users. This decision will ultimately be based on your security requirements rather than the ease of implementation. As a general rule of thumb, your Admin group should be given rights to all privileges for all access lists.
7. For this example, we will be applying security privileges first to our Admin group and then to our new user Kevin. Select the option **Group**.

8. Select your user group for administrators. In our example we will use **Admin** - a group created during the installation process.

9. Highlight all privileges listed in the Unassigned privileges list.

   Standard Window's mouse controls apply here. The easiest way to select all items in a list is to select the first item in the list - **getSteps** and then hold down the keys **Ctrl-Shift-End**. All items should be selected. Click the **left arrow** key to move all highlighted privileges to the Privileges allowed for user list.

10. Select the option **User**.

    When you do this a list of the current users defined on your system will be presented to you. A sample is shown below.
11. Select **Kevin** from the list of Users. You may need to scroll down through the list of Users to find the desired profile.

Next, you must determine what specific privileges you want this user to have. A list of the available authorities and their descriptions can be found later in this manual. Refer to the section on **Base Access List** for details on the available privileges.

12. For our example, we will be allowing our user Kevin to have the recommended rights for an Ordinary Workbasket User. Assign the privileges to our user by selecting the desired privilege from the list of Unassigned privileges and then clicking the right arrow to move the privilege to the Privileges allowed for user list.

Assign the following privileges to Kevin:

- `addWorkpieceItem`
- `addWorkpieceNote`
- `addWorkpieceValue`
- `advanceWorkpiece`
- `createWorkpiece`
- `getActiveWorkpieceList`
- `getClassList`
- `getWorkbasketList`
- `getWorkbasketTOC`
- `getWorkpieceClassColumnList`
- `getWorkpieceHistory`
- `getWorkpieceInfo`
- `getWorkpieceListByValues`
- `getWorkpieceNotes`
- `moveWorkpiece`
- `removeWorkpieceItem`
- `removeWorkpieceValue`
Your screen should look similar to the one shown below.

You have now successfully created an access list call AP which provides user Kevin with basic workbasket user privileges.

**Define Workpiece Class Details**

In Enterprise Workflow the Workpiece Class is used to define the type of information that will be maintained for the new item category. You can also define security rules for each workpiece class by selecting an access list, which includes a list of authorized privileges that will be associated with the workpiece class. In this example we will continue to work through our Accounts Payable example and define a workpiece class that will outline the key pieces of information that will be associated with each AP workpiece and limit user authorities through the use of the AP access list created early in the Getting Started chapter. Refer to the administrative section of this manual for details regarding workpiece classes.

Below are the instructions for defining a workpiece class.

1. **Login** to the Admin page of Enterprise Workflow.
   
   Refer to the Login to Enterprise Workflow instructions for help with this.
2. Click **Maintain Workpiece Class** from the list of actions on the left.

You will be presented with a screen similar to the one below.

![Initial Workpiece Class Screen](image)

3. Select **New** from the Class Name drop-down list.

You will be presented with a screen similar to the one shown below.

![New Workpiece Class](image)

4. Enter a **Class Name**, such as **PO**.
The value or name that you enter into Class Name will be the class name that will you see in all drop-down selection boxes throughout the product.

5. Select an AP from the Access List drop down list box for the new workpiece class.

Remember, access lists are used to control authority to various Enterprise Workflow privileges. It is important to note that you can assign a different access list to the workpiece class than you did to the user profile or the workbasket. This means that available privileges at any point in time will be the common privileges that are found among those assigned to the user profile, the workpiece class and the workbasket.

In this example, we will keep it simple and select the same access list throughout our sample exercises.

6. Enter a Description for the workpiece class, such as Purchase Order.

Your completed screen should look similar to the one below.

7. Click the Add button.

You will be presented with a screen similar to the one shown below.
8. Next, you must define the columns of information to maintain for each purchase order. Type **VendorName** in the Column List edit box.

Workpiece columns are attributes associated with the workpiece. When you create new workpiece items, you will choose the workpiece class, which in turn, will allow you to enter values for each of the available attributes defined in the workpiece class.

9. Click the **Add** button.

The new column will be shown and you will see several new buttons added to the screen. The Move Up, Move Down and Remove buttons can be used to change the order of the column names. The first three column names and the associated values for the workpiece will be included on most screens that list workpiece(s). You cannot have spaces or special characters in the Column Names of a Workpiece Class.

Your screen should look similar to the one shown below.
10. Continue adding the following columns:

PONumber
PODate
VendorNumber
POAmount

Your screen should now look similar to the one shown below.
You have now created a workpiece class PO from which you will be able to maintain five key pieces of information for each PO entered into the system. If you were to view active workpieces, the PO workpieces will display Vendor Name, PO Number and PO Date - the top three columns are always shown in the lists of workpieces. Use the Move Up and Move Down arrows to change the order of the columns.

Create a Workbasket

Workbaskets are similar in nature to your in box. Workbaskets act as holding tanks for items of work. Before you can create a workflow process you must first create any workbaskets that will be associated with the process. For example, if you are designing a process to manage accounts payable paper work (vendor invoices, purchase orders, receiving documents), then you may need multiple workbaskets: one for incoming AP documents, one for items requiring additional approvals or signatures and a third for approved invoices that need to be paid.

Below are the instructions to creating a workbasket.

1. **Login** to the Admin page of Enterprise Workflow.
   - Refer to the [Login to Enterprise Workflow](#) instructions for help with this.

2. Click **Maintain Workbaskets** from the list of actions on the left.
   - You will be presented with a screen similar to the one below.
3. Select --New-- from the Workbasket Name list box.

You will be presented with the following screen.

4. Enter a Workbasket Name, such as AP Purchase Orders.

5. Select AP from the Access List drop down list box.

Access lists are used to control security to the workbasket and the items contained within it.
For simplicity sake, we will be using the same access list for all sample exercises.

6. Click **Add**.

The new workbasket will be created. Your screen should look similar to the one shown.

7. Create another workbasket named **Special Approval POs**. Assign the Access List **Admin** to the workbasket.

   We will use this second workbasket to hold all purchase orders over a particular dollar amount as they will need another level of approval.

**Design a Workflow Process**

Each workflow process design is created and/or modified through Workflow Designer. Workflow Designer allows you to define all the intricacies of any business process that will be managed through Enterprise Workflow.

This exercise will help you create a simple purchase order approval process. Complete details on the use of Workflow Designer can be found in a later chapter in this manual. Workbaskets are similar in nature to your in box.
Below are the instructions to create a purchase order approval process.

1. Open the application **Workflow Designer** from program group **RJS Enterprise Workflow**.

   You will be presented with a Login screen similar to the one shown below.

   ![Workflow Designer Login Screen](image)

2. Enter a **valid Workflow User Name**, **Password** and **IP address**.

   The Enterprise Workflow installation process creates a user **Administrator/Administrator** (UserName/Password) that you can use. The Administrator user profile has full administrative authorities within Enterprise Workflow. In general, the IP address will consist of either a DNS name or IP address, a colon and then the IP Port number. The IP port number is assigned during the Tomcat installation; the default value is 8080. Below is a sample:

   10.1.1.6:8080  
   Where 10.1.1.6 represents the server IP address and 8080 represents the IP port number.

3. Click **OK**.

   You will be presented with an opening screen like the one shown below. All new processes begin with Start and Stop task markers.

   ![Workflow Designer Opening Screen](image)

4. Select **Workbasket** from the Insert menu. Select **Constant Name** from the submenu.

   The Workbasket submenu asks for either a Constant Name or a Variable Name. Use the
constant name if this step in the process represents one specific workbasket. The Variable Name workbasket is helpful if your process includes multiple workbaskets, for example several people that can do the processing at this step in the workflow.

A new Workbasket will be placed in the upper-left corner of screen as shown below.

5. **Move your cursor over the new Workbasket object.**

   When you do this, the cursor will change to a hand and sizing handles will appear around the Workbasket object.

6. **Move the cursor over the sizing handle in the center of the object.** The cursor will again change, this time to a directional cross-hair which signifies you can move the object. Then **click and drag the object down and to the right** of the Stop object.

   Your screen will look similar to the one shown below.
7. **Right-click** over the workbasket.

   After a few seconds you will be presented with a list of workbaskets currently available for use.

   ![List of Available Workbaskets](image)

8. Select **AP Purchase Orders** from the list and click **OK**.

   You will now see AP Purchase Orders, or at least part of the text, on the front of the workbasket.

9. Select **Decision Point** from the Insert menu.

   A decision point marker will be inserted in the upper left corner of the Designer window.

10. Drag and drop the decision point marker so that it is **below and to the right** of the AP Purchase Orders workbasket.

   Your screen should look similar to the one shown below.

   ![New Decision Point Step Marker](image)

11. Again, add **Constant Name Workbasket** to the workflow. Move the workbasket down and to the right of the Decision Point marker.

   Your screen should look similar to the one shown below.
12. Assign **Special Approval POs** to the new workbasket.

13. Drag and drop the **Stop marker** to below and to the right of the Special Approval POs workbasket.

14. Select the connector (arrow) that is between Start and Stop. **Click on the arrow-end of the connector and drag the end to the Purchase Orders workbasket.**

By doing this, you are placing the Purchase Orders workbasket as the first step in the process. All workpieces assigned to this workflow process will initially be placed in the Purchase Orders workbasket.

Your screen will look similar to the one shown.
15. **Click and drag the mouse from the Purchase Orders workbasket to the Decision Point.**

A connector will be drawn between the two step markers and then a Change Option dialog box will be presented. The Change Option dialog can be used to name an action associated with the advancement of the workpiece in the workflow or to force a comparison, such as at a Decision Point, that determines where the workpiece moves next in the workflow process.

![Change Option Dialog](image)

16. **Type Approved in the Option edit box and click OK.**

We are using the dialog box to name the action. This will be helpful to those processing purchase orders as they will see the action Approved.

17. **Draw a connector between the Decision Point and Stop.**

The new connector will be titled DEFAULT. The first connector created from a Decision Point will always be named DEFAULT. You must have a default path that the decision point can use when all other connector comparisons evaluate to false.

18. **Select Curved from the Arrow Style list box found on the tool bar. Click and drag one of the sizing handles in the middle of the connector to the left.**

This will create a curve in the line so that it does not lay directly over our second workbasket. Curving the line is strictly for aesthetics and makes it easier to read the Option detail associated with the connector.

Your screen will look similar to the one shown below.
19. Draw another connector **between the Decision Point and the Special Approval POs workbasket**.

20. Type **POAmount > 10000** in the Option edit box.

   This comparison will look at the POAmount value, a column name associated with the PO Workpiece Class, and advance any purchase order where the amount is greater than 10,000. The column name must be typed exactly as it is entered in the workpiece class. You cannot have spaces in the column names. Refer to **Decision Point Expressions** section of the manual for additional help creating comparisons in the Change Option dialog box.

   A sample of the Change dialog is shown below.

21. Again, to help you be able to read the Option details for the connector, **click and drag the middle sizing handle to the right** which will curve the connector.

   Your screen will look similar to the one shown.
22. Create one final connector between the Special Approval Pos workbasket and Stop.

23. Select Save from the Process menu.

You will be presented with a screen that allows you to name your new workflow process.

24. Type Purchase Orders in the process ID.

In general, the name you enter into Process ID is the name that is surfaced in most workflow process selection list boxes. The Process Name and Process Description can be used to offer additional descriptions to help users and flow designers understand what and how this process will be used.

25. Type any additional descriptions into Process Name and Process Description edit boxes.

Your screen will look similar to the one shown below.
26. Click **OK** to save the process.

You will receive a message stating the changes were saved successfully.

You have now created your first workflow process! We will use this process in the next couple of sections of the Getting Started section.

### Create/Advance Workflow Workpieces

Creating and advancing workpieces are activities that most of your Enterprise Workflow users will perform often. By creating a new workpiece, we are essentially saying we have a new work item that needs to be inserted into one of our business processes. Once a workpiece is inserted into the process, the workpiece will move from workpoint to workpoint within the process. Sometimes the workpiece is advanced by a user (our sample will show you how) and other times the workpiece will be advanced automatically through a decision point or an exit program.

This section describes how you can create a workpiece using the Admin JSP page and then advance the workpiece using the standard Workbasket JSP page. In this exercise we will create a workpiece that will advance through the AP Workflow Process designed in the section Design a Workflow Process. When creating a new workpiece we will assign values to workpiece column names and enter a fictional reference to a document repository.

**Below are the instructions to creating a workpiece within the PO workflow.**

1. **Login** to the Admin page of Enterprise Workflow.

   Refer to the Login to Enterprise Workflow instructions for help with this.

2. Click **Create Workpiece** from the list of actions on the left.

   You will be presented with a screen similar to the one below.
3. Select **PO: Purchase Orders** from the Class drop-down list.

4. Select **Purchase Orders** from the Process drop-down list.

5. Type **Test PO item** in the Description edit box.

   The description can help further define this particular workpiece within the class it is associated with.

6. We will leave the Priority at **50**.

   Priority can be set to any value between 1 and 99, 1 being the highest priority. Priority can be used to assign an importance to the processing order within any workflow process.

   A sample of the screen so far is shown below.
7. Type `m:\Document Repository\PO12345.tif` in the URL Edit box.

   If you are using RJS WebDocs with Enterprise Workflow, you would place the WebDocs document link in the URL field.

   The value we have entered above is an arbitrary value to show you how to use the URL field. This is not a valid URL to use to view the associated purchase order image.

8. Type **Purchase Order # 12345** in the Description edit box.

   Description is used to describe the document linked to this workpiece.

   In our example we will not be using Type. Type can be used in conjunction with Collection Points. If you have collection points defined within your workflow process the Type field is used as a reference type.

9. Click the **Add** button to insert the reference into the workpiece.

   The Workpiece Reference information is stored and displayed on the page. Your screen should look similar to the one below.
10. Scroll down so that you can see the Workpiece Attributes section of the page. Type **POAmount** into the Name edit box.

This is the column name as it was typed in the Workpiece Class. It must be entered exactly the same as it is stored in the Workpiece Class definition.

11. Type **15000** into the Value edit box.

The 15,000 represents the total amount of the purchase order. We will use this value to determine how the workpiece advances through the purchase order process.

12. Click the **Add** button to insert the column value into the workpiece attribute section.

The attribute value is stored and displayed on the page. Your screen should look similar to the one below.
13. Add **PONumber** of **12345** as a second attribute value.

The attribute and value will be listed below **POAmount**.

14. Click **Create** button to create the new workpiece item with the entered references and attributes.

The workpiece is now moving through the PO workflow process. It currently resides in the Purchase Orders workbasket waiting for someone to do the necessary processing.

Next you will advance the workpiece to the next step in the process.

**Advance Workpiece**

1. Open your browser and enter the **URL** to access the Enterprise Workflow server. Use the IP address and port number that was used during the configuration of Enterprise Workflow. A sample URL is shown below.

   ![URL](http://10.1.1.10:8089/WorkflowWebServiceClients/WorkflowWebserviceClients/Workflow.jsp)

   The above URL will take you to the default end-user interface page of Enterprise Workflow.

   You will be presented with a screen similar to the one shown below.
2. Select **Login** from the Functions list on the left.

   Login using our sample user/password of **Kevin/Kevin**.

3. Click on the **Workbaskets** tab.

   You will then be presented with a list of functions and workbaskets that our user Kevin has access to.
4. Click on **AP Purchase Orders** under the Workbaskets section.

You will be presented with a list of active workpieces in the AP Purchase Orders workbasket. Currently we have just one workpiece in the workbasket.

5. Move the cursor **over the Test PO Item workpiece and click**.

You will be presented with a detail screen for the workpiece.
If a person were actually processing this purchase order, they would most likely click on the link to the document reference and view the purchase order. Since we have placed a sample URL into the link, we will not actually complete the task of viewing the purchase order. In addition to viewing the reference documents, the processing clerk may also change or add new column data such as vendor number. Since this activity works the same in the workpiece creation screen and this screen, we will not make any changes or additions to index values.

6. Notice that Approved is listed in the Action list box. Click Go to advance the workpiece to the next step of the process.

Since our POAmount is greater than 10,000, the workpiece will be advanced to the Special Approval POs workbasket. If the POAmount was less than 10,000, then the workpiece would have advanced to the end of the process.

You will be returned to the workbasket list which will be empty since we had just one workpiece in it.

**Find a Workpiece**

Periodically, it will be necessary for your users to search for a particular workpiece and review the workflow history of the item. There are two main ways in which you can view the details associated with a workpiece: view all active workpieces or search by column name and value. This section will step you through both processes and show you how to find a particular workpiece, and then review the activity for the workpiece once found.

**Below are the instructions to view all active workpieces**

1. **Login** to the Admin page of Enterprise Workflow.

   Refer to the [Login to Enterprise Workflow](#) instructions for help with this.
2. Click **Active Workpiece List** from the list of actions on the left.

The system will retrieve a list of all active workpieces, for all workflow processes, and then display the entire list. You will be presented with a screen similar to the one below.

![Active Workpiece List](image)

3. Scroll through the list and **click on the Purchase Order workpiece** which will most likely be the last item on the list.

The history of the workpiece will be displayed. Included in the history will be each workflow transaction as the workpiece moves from one step to the next. Transaction detail will show the date and time, the user that completed the task, before and after step details and workpiece processing priority. A sample of the screen is shown below.
Workpiece History From the Search Screen

Below are the instructions to search for a specific workpiece

1. **Login** to either the Workbasket or Admin pages of Enterprise Workflow.

   Refer to the [Login to Enterprise Workflow](#) instructions for help with this.

   Workpiece Search can be completed from both the Admin and the Workbasket pages. The workpiece search works exactly the same in both interfaces.

2. **Click Search for Workpiece** from the list of actions on the left.

   You will be prompted to enter Name and Values for the search. Name is the Column Name as defined in the Workpiece Class and value is the actual value entered into the workpiece. For our example we will use the PONumber for our search. Keep in mind, you may enter as many search criteria as you would like but all criterion must be true for a workpiece to be returned from the search.

   The Search Workpiece screen will look similar to the one below.
3. Type **PONumber** in the Name edit box.

   This is the Workpiece Class Column Name. The name must be typed exactly as stored in the workpiece class definition and is case sensitive.

4. Type **12345** in the Value edit box.

   This is the actual PO number we are searching for.

5. Click the **Add to Search** button.

   The new criterion will be listed on the screen as shown below.
6. Click the **Search** button.

If multiple people are testing or running through the getting started exercises, you may see multiple workpieces that match the PO Number of 12345 in your search results list.
7. Scroll through the list and **click on your Purchase Order workpiece** which will most likely be the last item on the list.

The history of the workpiece will be displayed. Included in the history will be each workflow transaction as the workpiece moves from one step to the next. Transaction detail will show the date and time, the user that completed the task, before and after step details and workpiece processing priority.

**Create User Group**

User Groups are especially helpful in eliminating the need to individually define security access for each user account. Often users and their associated security restrictions can be categorized, which allows you to define security access for the entire group. This section steps you through the creation of a user group and insertion of the appropriate users into the group.

1. Log into the Enterprise Workflow Admin page using a user profile that has administrative authorities.

   During the installation process of Enterprise Workflow the user **Administrator** is created and given full rights to all Enterprise Workflow functions. The default password for **Administrator** is **Administrator**.

2. Click **Maintain User Group** from the list of actions on the left.

   You will be presented with a screen similar to the one below.

3. Select **New** from the User Group Name drop down list.

   You will be presented with a screen similar to the one shown below.
4. Type **AP** for the User Group Name.

The User Group Name value is the group name that will be displayed in most list boxes with Enterprise Workflow.

5. Enter **Accounts Payable** as the Description for the user group.

Your screen will look similar to the one shown below.
6. Click the **Add** button.

You will be presented with a screen similar to the one below.

7. From the list of **Non-Members** select a user that you want included in the new group.

   For our example, we will use our newest user, **Kevin**.

8. Click on the **left arrow** button to move the user Kevin to the Members list.

   Your screen should look similar to the one below.
9. Continue to select any additional users from the non-members list and move them to the Members list until you have added all the required users to the group.

The new user group has been created. You can return to this screen at any time to add or remove users from the group. Link the group to access lists and workbaskets to simplify your security definitions rather than defining privilege authorizations for individual users.
Operations

Overview

RJS Enterprise Workflow is a highly flexible product for building and running business process management processes.

There are three principle components: The Workflow Designer, Workflow Engine and the User Interface.

RJS Workflow Designer is a windows application that accesses builder methods of the workflow engine as users graphically draw each workflow process.

The Workflow Engine is a set of web services, java classes, and database tables that maintain state and move each workpiece from point to point in a process.

The User Interface is a set of Java Server Pages (JSP) that provide the human contact. The user interface can be divided into two sub-components, the Administrator Tasks, and the Workbasket User Tasks.

Process Elements

The workflow engine moves workpieces from one process element or step to the next according to a predefined process. The actual step the workpiece might take is determined dynamically by the contents of the workpiece and by actions the user might choose at an interface point.

The process is defined by a designer using the RJS Workflow Designer tool. The designer places workflow elements in the process and connects them. Some elements allow options; that is, more than one connector leading from the element. The workflow engine evaluates the range of options and chooses the appropriate one to conform with the process designer's intentions.

These are the process elements supported by the RJS Workflow Designer and the workflow engine:

- **Start Point**
- **Stop Point**
- **Work Point**
- **Decision Point**
- **Exit Point**
- **Collection Point**

Start Point

The start point represents the place where new workpieces enter the workflow. When a new workpiece is created, the workflow engine creates a timestamp in the workpiece history for the start point and immediately advances the workpiece to the next process element in the workflow. A process can have only one start point. There can be only one connector leading from a start point and no connectors can point to a start point. The one connector leading from a start point has no options.
Stop Point

When the workflow engine advances the workpiece to a stop point it stops workflow processing. The workpiece will not be advanced from a stop point. A stop point may have many connectors leading to it. A stop point may not have any connectors leading from it. A process may have many stop points, but must have at least one.

Work Point

A work point is an external interaction point. When the workflow engine advances a workpiece to the work point it stops advancing the workpiece. An external agent (often a user but not always) must access the workpiece, do any work that it needs to do and then signal the workflow engine to continue advancing the workpiece.

When the workflow engine advances the workpiece to a work point, it assigns the workpiece to a workbasket. The process developer chooses one of several methods of assignment when he adds the work point to the process diagram. Workbaskets have names. The workflow engine can be asked to produce a list of all the workpieces that have been advanced to a workbasket with a particular name. The workbasket name must be defined to the workflow system before it is used in a process.

Any number of connectors may lead to or from a workbasket. Each connector leading from a workbasket has a unique option name. A list of the option names is presented to the agent when he displays the workpiece. The agent must choose one of those options. The workflow engine will then advance the workpiece along the connector associated with that name.

The external agent "working" a workbasket may be human or an external process. If it is a human, the human is probably taking some action prompted by the workpiece arriving in the workbasket. The human may alter the workpiece in some way before dispatching it by choosing an option. For instance he may add or alter one of the attributes of the workpiece. Or he may alter the electronic document to which the workpiece refers. He may do none of these things and take some external action, like keying information into another system based on the contents of the electronic document.

Here's how a work point looks to the process designer. Both of these work points are statically linked to named workbaskets.
In this example, there exists an external order system. A user must look at the document and key some values into the order system. Once the order has been entered, this process is done. Some orders need artwork, however. If the order needs artwork, the order entry agent will choose the option to send the workpiece to the art department. The art department creates the artwork and sends the workpiece back to the order entry department. Before they do so, they will add an attribute to the workpiece stating the location of the artwork (they may even attach the artwork to the workpiece as an electronic artifact).

**Decision Point**

When a workpiece reaches a decision point, the workflow engine evaluates the option paths leading from the decision point in the process diagram. Each path has an associated Boolean expression. The workflow will advance the workpiece along the first path in the evaluation order that is true. There must always be a default path from a decision point. If no other paths have expressions that are true, the default path is followed. A workpiece never remains at a decision point.

**Exit Point**

An exit point represents a server extension program function. When the workflow engine advances a workpiece to an exit point, the java class whose name is associated with the single option leading from the exit point on the process diagram is called. The string value also associated with the option is passed to that java class along with information about the workpiece.

**Collection Point**

A collection point is where a workpiece "waits" for a reference of a particular type to be added to it. When the workflow engine advances a workpiece to a collection point it examines the options leading from the collection point on the process diagram. Each option is associated with a reference type. If the workflow determines that the workpiece already contains a preference of the given type it will advance the workpiece along that path. Otherwise it stops advancing the workpiece. Each collection point has a timeout value expressed in years, months, days, hours, minutes and seconds. When the workpiece has been at the collection point for the designated amount of time, it will follow the timeout path defined on the process diagram.

If a reference is added to the workpiece while it is still at the collection point (no timeout yet), the workflow engine will examine the paths leading from the collection point on the process diagram and look for a type matching the new reference. If one is found, the workpiece is advanced along that path.

**Workpiece**

A workpiece is the fundamental unit of workflow. Each workpiece is on one process, has references to files, documents or other elements accessible through a URL, and has named attributes. A workpiece is also an instance of a workpiece security class. The class specifies the security model for the workpiece and describes how the workpiece will be displayed in workpiece lists such as workbaskets.

**Process**

Each workpiece is associated with a process.
Priority

Workpieces are listed in workbasket tables of contents in order of their priority. Priority is an alpha value that ranges from "00" to "99". The priority may be changed by a user or by an exit point.

Actions

A workpiece has actions associated with it. These actions vary by where the workpiece is currently located in the process. The actions at a work point match the options in the process diagram leading from the work point. The user chooses an action when he advances the workpiece from the work point.

Notes

Users may enter notes on a workpiece. Each note is labeled with the user's ID and time-stamped.

Attributes

A workpiece can have an infinite number of attribute values. Attributes can be added when the workpiece is created or by user's or processes anywhere along the process. An attribute consists of a name and a value. An attribute is usually associated with a single fact about a workpiece. Attributes can be used to influence the workpiece path at decision points in the workflow, can be used when assigning the workpiece to a workbasket at a work point, or can be used by exit points. Attributes and their values are also used to search and find workpieces.

References

References are special unnamed attributes that "point" to an external object in a repository. In effect, a reference is an HTML anchor tag including a universal resource locator (URL). When a reference is displayed in a browser context it will display as a hyperlink. When the hyperlink is clicked the object will be displayed or otherwise handled according to the object type and the instructions in the hyperlink.

When a reference is added to a workpiece, it is given a type. This type is used at collection points in the process diagram.

References may be added to a workpiece when it is created, or by a user at a work point, or by a program using a workflow API. When a reference is added to a workpiece, the workpiece position within the workflow is checked. If the workpiece is waiting in a collection point, the workpiece may be advanced.

Login

Enterprise Workflow includes two JSP web pages that can be used to log into the workflow server: Admin page and the general Workbasket page. The Admin page is used to access all administrative features of Enterprise Workflow. Typically a user would need administrative authority to use the features found on the Admin page. The general Workbasket page will be used on a daily basis by your users to attach reference material to a workpiece, add notes and advance the workpiece to the next step in the process.

Admin: The default format for the Admin URL is as follows:
The IP Address is the IP address where you have installed Enterprise Workflow. Port number is the port number associated with your web application server (i.e., Tomcat or WAS). A sample URL to access the Admin page might look like the following:

http://10.1.1.6:8089/WorkflowWebServiceClients/WorkflowWebserviceClients/Admin.jsp

General Workbasket: The default format for the URL is similar to the used for Admin.

http://10.1.1.6:8089/WorkflowWebServiceClients/WorkflowWebserviceClients/Workflow.jsp

The IP Address is the IP address where you have installed Enterprise Workflow. Port number is the port number associated with your web application server (i.e., Tomcat or WAS). A sample URL to access the general Login page might look like the following:

http://10.1.1.6:8089/WorkflowWebServiceClients/WorkflowWebserviceClients/Workflow.jsp

**Workflow Designer**

RJS Workflow Designer is a Windows-based graphical designer for creating workflow processes. A workflow process is a graph with nodes and paths. Workflow Designer lets you manipulate the graph according to simple rules for workflow.

Workflow steps are represented as nodes on the graph.

**Connectors**

Connectors are directed paths that lead from step to another step. Some connectors have values associated with them. What the values mean is dependent on where the connector originates.

**Work Point Options**

The connectors originating from a work point have a name associated with them. The name indicates an option the user must choose when advancing the workpiece. The user need not know where the workpiece goes next in the process. Only that the action (name) is appropriate for the task he has performed. For example, in a purchase order approval scenario, you find actions such as Done, Approved, or Re-Evaluate.

A sample of a named action associated with a Work Point is shown below. In this example the action name, found in the Option field, is named Approved.
Decision Point Expressions

Decision Point option connectors have associated Boolean expressions in java syntax. These expressions are entered into the Option field on the dialog box.

The evaluation order from a decision point is important. The workflow engine evaluates each path from a decision point in numerical order using the values you enter into the Evaluation Order field. It will stop evaluation as soon as it finds an expression that is true and advance the workpiece along that path. Below shows a sample of the decision point connector that has been labeled as the first option to evaluate.

You can have unlimited decision point connectors. However, you will need to maintain these connectors so keep that in mind as you define the process.

Decision points must always have a default path leading from them. If no expression is found that evaluates to true, the default path is taken. You might think of the default path as having an expression that always evaluates to true. The first Decision Point connector that you draw will be assigned an evaluation order of 999 and will be defined with an option of DEFAULT. In the example below, the Evaluation Order value is set to 999, a value far beyond the expected number of connectors for this decision point.
All expressions entered into the Option field must evaluate to either true or false and must be entered using Java Boolean syntax. Expressions can be as simple as comparing a variable value to a literal value or you may choose to use any available Java string methods. Below is a table that displays common comparators and a description of each that are available to use in your expressions:

<table>
<thead>
<tr>
<th>Comparator</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&gt;</td>
<td>Greater Than</td>
</tr>
<tr>
<td>&lt;</td>
<td>Less Than</td>
</tr>
<tr>
<td>&lt;=</td>
<td>Less Than or Equal to</td>
</tr>
<tr>
<td>&gt;=</td>
<td>Greater Than or Equal to</td>
</tr>
<tr>
<td>==</td>
<td>Equals</td>
</tr>
</tbody>
</table>

There are many Java string methods that can also be used as part of your Boolean expressions in the Option field. Below is a table of some of the more common methods available for your use:

<table>
<thead>
<tr>
<th>Comparator</th>
<th>Description</th>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>compareTo(string)</td>
<td>Compares two strings lexicographically.</td>
<td>Integer value of:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>match</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-1 or 1</td>
</tr>
<tr>
<td>compareToIgnoreCase(string)</td>
<td>Compares two strings lexicographically, ignoring case differences.</td>
<td>Integer value of:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>match</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-1 or 1</td>
</tr>
<tr>
<td>contentEquals(StringBuffer)</td>
<td>Returns true if and only if this string represents the same sequence of characters as the specified StringBuffer.</td>
<td>Boolean value:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>True</td>
</tr>
<tr>
<td></td>
<td></td>
<td>False</td>
</tr>
</tbody>
</table>
## Comparator Description

<table>
<thead>
<tr>
<th>Comparator</th>
<th>Description</th>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>endsWith(String)</code></td>
<td>Tests if this string ends with the specified suffix.</td>
<td>Boolean value: True</td>
</tr>
<tr>
<td><code>matches(RegEx)</code></td>
<td>Tells whether or not this string matches the regular expression (RegEx).</td>
<td>Integer value of: 0</td>
</tr>
<tr>
<td><code>trim()</code></td>
<td>Trims leading and trailing spaces from the given string.</td>
<td>Returns: copy of the string with leading and trailing spaces removed.</td>
</tr>
</tbody>
</table>

Refer to [http://java.sun.com/j2se/1.4.2/docs/api/java/lang/String.html#method_summary](http://java.sun.com/j2se/1.4.2/docs/api/java/lang/String.html#method_summary) web site for a full list of available string methods and complete syntax documentation.

Sample expressions you might find in the Option field are shown below:

String Comparison: `State.compareTo("OH")==0`  

or  

Numeric Comparison: `DepartmentNo==45`  

or  

Boolean Expression: `ProdID.contentEquals("XYZ001234")`

## Exit Point Class Definition

The one and only path leading from an exit point has associated with it a class name and an option string. The class name names a java class available to the workflow engine that implements the "WorkflowExitInterface" java interface. The option string is passed, as a string, to the "go" method of the class. Ask the author of your java class for instructions as to the contents of this field.

The following dialog will be used to define the Java class to be called by the Exit Point.

![Exit Point - Java Class to Call](Exit_Point_Java_Class_to_Call.png)
Collection Point Reference Type

Collection Point is used as a holding tank for workpieces that require additional documentation before the workpiece can advance in the process. For example, an insurance company that is processing hospital claims may use a collection point to hold all claims that are waiting for surgical reports. Workpieces do not advance in the process until the appropriate document is added as a reference to the document.

The options leading from a collection point indicate what to do when a certain type of reference is present in or added to the workpiece.

There must always be a timeout path leading from the collection point and the collection point itself has a timeout value associated with it. If the workpiece is still held at the collection point when the designated period of time has passed, the workflow engine will advance the workpiece along the timeout path.

Syntax of the timeout value is as follows:

```
IntegerPeriodIdentifier[IntegerPeriodIdentifier . . .]
```

Where

- **Integer** = represents the numeric value of time to wait.
- **PeriodIdentifier** = represents a character string that represents the desired time period. Available periods are:
  - **Year**: Can be identified with any portion of the string "Year" since the initial character "Y" is unique to this time period. E.g., "Y", "Ye", or "Year" can all be used to identify the period as Year.
  - **Month**: Must be identified with at least the first two characters of "Month" so that the time period can be distinguished from Minutes. E.g., "Mo", "Mon", or "Month" can all be used to identify the period as Month.
  - **Day**: Can be identified with any portion of the string "Day" since the initial character "D" is unique to this time period. E.g., "D", "Da", or "Day" can all be used to identify the period as Day.
  - **Hours**: Can be identified with any portion of the string "Hours" since the initial character "H" is unique to this time period. E.g., "H", "Ho", or "Hours" can all be used to identify the period as Hours.
  - **Minutes**: Must be identified with at least the first two characters of "Minutes" so that the time period can be distinguished from Month. E.g., "Mi", "Min", or "Minutes" can all be used to identify the period as Minutes.
  - **Seconds**: Can be identified with any portion of the string "Seconds" since the initial character "S" is unique to this time period. E.g., "S", "SE", or "Seconds" can all be used to identify the period as Seconds.

Integers and Period Identifiers can be used in combination for more flexibility. Some sample timeout values are shown below:

- 1D Represents one day
- 2Mo Represents two months
2H30Mi   Represents two hours 30 minutes

An example of a timeout value is shown below.

![Change Option dialog]

**Process Menu**

**Retrieve**

Choose retrieve to retrieve an existing process definition from the workflow engine. If the process was saved on this Windows machine, the previous locations of the steps, arrows and their paths will also be retrieved. Otherwise, Workflow Designer will attempt to lay out the steps in a logical fashion.

You will be prompted to select from a list of processes known to Enterprise Workflow.

![Select Process to Open dialog]

**Save**

Choose Save to save the current process in the designer.

**Save As**

Choose this option to save the current process with a new name. The process stored under the previous name will not be changed. You will be prompted to enter a new name for the process.
New

Create a new process. Workflow Designer will create a new minimal process with a start point, a stop point, and a connector originating at start and terminating at stop. A sample of a new process is shown below.
It is important to note that if you are currently working on a process description and you have not saved changes, you will lose your work when you choose New from the Process menu. You will not be prompted to save current work.

**Insert Menu**

**Decision Point**

When selected, a new Decision Point will be placed on the work surface in Workflow Designer. Decision points are used to compare attribute values associated with the workpiece with pre-defined conditions associated with Decision Point Expressions. Each decision point must have a default path that will be used when all other expressions evaluate as false.

Refer to the [Decision Point Connectors](#) section for details on creating expressions.

**Workbasket**

By selecting Workbasket from the Insert menu you can create a work point in the process. The purpose of a work point is to provide a place-marker in the workflow process where a user can view and possibly perform necessary tasks associated with the current step in the process. When, during processing, a workpiece arrives at a work point it is assigned to a specific workbasket. It is from this workbasket that a user will access the workpiece and complete work associated with this step before advancing the workpiece to the next step in the process.

When you insert a workbasket into the workflow design you will be prompted to select either a Constant Name or Variable Name workbasket. When Constant Name is selected, you will choose from the list of current workbasket to assign to this step in the process. If you choose Variable Name, then you will enter a Variable Name for the workbasket - Variable Name value will be a Column Name associated with the workpiece class. The value of that variable must match a pre-defined workbasket.

Constant Name workbasket do not have restrictions as the number of users that can access the workpieces within the workbasket. However, users must have authority to the workbasket before they will be allowed to view or change the contents of a workbasket.
Variable Name workbaskets can be used when there are multiple individuals that may process work at a given work point but each require their own personal workbasket. In this case, workbaskets can be generated for each individual that will use the process. In essence, this will create virtual workbaskets for each individual defined for the step. In contrast, if you create a workbasket with a constant name than just one workbasket is created for the current process step.

**Exit**

Use the Exit menu option to insert an Exit Point into the process definition. Remember, an exit point is used to process the workpiece programmatically.

**Collection Point**

By selecting Collection Point from the Insert menu you can insert position in the business flow where workpieces can be accumulated. Collections points are especially helpful if there are times in the process where you may need place workpieces into a holding bin while you wait for additional information. For example, in a purchase order process you may approve the purchase but the workpiece needs to be held in a wait-state until the vendor sends an invoice for the approved purchase.

**Stop**

Selecting Stop from the insert menu will create an end point for the process. Your defined processes may contain multiple stop points.

**Web Service Menu**

Selecting the Web Service Menu item will allow the user to log into Enterprise Workflow while in Workflow Designer. This menu option is helpful if you have lost your connection to the Enterprise Workflow server and need to reestablish your connection or if you need to log in as a different Enterprise Workflow user.

The Workflow Designer Login dialog box will look similar to the one shown below.

The IP Address consists of either an IP address or DNS server name, a colon, and the IP port number associated with your web application server (i.e., Tomcat or Websphere).

**Workbasket User Interface**

The Workbasket User Interface can be used to process workpiece items within Enterprise Workflow. This interface provides basic functionality to the common users to be able to create new workpiece items, process existing workpiece items and to search for active workpieces. The number of workbaskets included in the list on the left will be based on those workbaskets the user has authority to.
A sample of the Workbasket Interface is shown below.

![Workbasket User Interface](image)

Each function provided in the Workbasket Interface has been covered in other portions of this manual. Below are section links to each of those functions should you need additional information.

- Login
- Create Workpiece
- Search for Workpiece
- Working with Workbaskets

**Implementing Workflow Security**

The workflow security model is quite flexible. This flexibility imparts some complexity in building your security implementation.

Each asset to be secured is associated with an access list. An access list is a list of users or a group of users. Each user or group in the access list is associated with set privileges for that access list. The privileges restrict the actions the user may take on the asset secured by the access list.

Each user profile is also associated with an access list. The user is granted permissions through this access list to use functions that are not associated with an asset - workbasket sand workpieces. The user cannot be given more privileges through an access list associated with an asset than he is granted through the asset list associated with his user profile.

The assets that are secured by access lists are workbaskets and classes. Each workpiece is a member of a class. So workpieces are secured by access lists through their class.
There is a privilege for each of the workflow web service methods. Granting a privilege to a user gives the user authority to use that method.

If the method involves more than one asset (for instance, advancing a workpiece that is in a workbasket) the user must have privileges to that method for all assets.

A user may belong to more than one group. A user may be granted a privilege through any of those groups.

Specifically, any action in Enterprise Workflow can be restricted through the assigned privileges associated with the user, workbasket and the workpiece (through the workpiece class). For an action to be allowed the privilege must have been granted to the user, the current workbasket and the current workpiece. If the necessary privilege has not been assigned to any one of the three categories your desired action will fail. The Venn diagram shown below illustrates how the assigned access list privileges must intersect with the privileges of the other assets to successfully complete the task at hand.

Access List Privilege Convergence

If a user is included by user name in an access list, the user will not be granted privileges through any group to which they belong that is also included in the access list.

Notice that the documents, files, and objects that may be referenced by a workpiece are secured by their own repository. See the rules for the repository you use; for example WebDocs - iSeries Edition.

Base Access List

Most installations will associate user profiles with a base access list where users are assigned privileges by groups.

The same access list is associated with all the profiles. The access list includes only user groups. The user is assigned to one of these groups and thereby assumes the privileges associated with the group.
A user can never gain more privileges than are assigned through the access list associated with his user profile.

Refer to the [Maintain Access List](#) section of the manual for a description of each privilege.

Here is a suggested setup for a base access list. Group names are across the top. Ordinary workbasket users have only the authority they need to manipulate workpieces within workbaskets. Supervisors can also change the priority of a workpiece or stop a workpiece. Help Desk has the privileges needed to set up new users, workbaskets and access lists, but no authority to work with the items in a workbasket. Members of the Process Designer group have the authority they need to create and alter processes. The Administrator group have privileges to use any method.

<table>
<thead>
<tr>
<th>Ordinary Workbasket User</th>
<th>Workbasket User/Supervisor</th>
<th>Help Desk</th>
<th>Process Designer</th>
<th>Administrator</th>
</tr>
</thead>
<tbody>
<tr>
<td>addWorkpieceItem</td>
<td>addWorkpieceItem</td>
<td></td>
<td>addWorkpieceItem</td>
<td></td>
</tr>
<tr>
<td>addWorkpieceNote</td>
<td>addWorkpieceNote</td>
<td></td>
<td>addWorkpieceNote</td>
<td></td>
</tr>
<tr>
<td>addWorkpieceValue</td>
<td>addWorkpieceValue</td>
<td></td>
<td>addWorkpieceValue</td>
<td></td>
</tr>
<tr>
<td>advanceWorkpiece</td>
<td>advanceWorkpiece</td>
<td></td>
<td></td>
<td>createAccessList</td>
</tr>
<tr>
<td>createAccessList</td>
<td>createAccessList</td>
<td></td>
<td></td>
<td>createClass</td>
</tr>
<tr>
<td>createClass</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>createGroup</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>createUser</td>
<td></td>
<td></td>
<td></td>
<td>createGroup</td>
</tr>
<tr>
<td>createWorkbasket</td>
<td>createWorkbasket</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>createWorkbasket</td>
<td>createWorkbasket</td>
<td></td>
<td></td>
<td>createWorkbasket</td>
</tr>
<tr>
<td>createWorkpiece</td>
<td>createWorkpiece</td>
<td></td>
<td></td>
<td>createWorkpiece</td>
</tr>
<tr>
<td>deleteClass</td>
<td>deleteClass</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>getActiveWorkpieceList</td>
<td>getActiveWorkpieceList</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>getActiveWorkpieceList</td>
<td>getActiveWorkpieceList</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>getClassList</td>
<td>getClassList</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>getClassList</td>
<td>getClassList</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>getPrivilegeList</td>
<td>getPrivilegeList</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>getPrivilegeList</td>
<td>getPrivilegeList</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>getProcessList</td>
<td>getProcessList</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>getProcessList</td>
<td>getProcessList</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>getSteps</td>
<td>getSteps</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>getUserGroupList</td>
<td>getUserGroupList</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>getUserGroupList</td>
<td>getUserGroupList</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>getUserGroupMemberList</td>
<td>getUserGroupMemberList</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>getUserGroupMemberList</td>
<td>getUserGroupMemberList</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>getUserList</td>
<td>getUserList</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>getUserList</td>
<td>getUserList</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Create a New User

When you create a new user, associate the Base Access List with his user profile and him to the appropriate group. Refer to the section Create User Profile in the Getting Started chapter for a step by step guide to creating a user profile in Enterprise Workflow.

Create a Private User Workbasket

You may want to build a workbasket such that only one user, their supervisors, and the administrators can work with items in the workbasket. This is the suggested setup:

1. Create a user profile.
Associate the base access list with the user profile.

2. Create an access list with the same name as the user's ID.

3. Give the user specific privileges in that access list.
   In general you should assign the same privileges here that the user has been granted through the base access list by way of a group.

4. Give appropriate privileges to the supervisor and administrator groups.

5. Create a workbasket with the same name as the user ID and secure it with the access list created above.

**Administrative User Interface**

The Administrative User interface can be used to manage all aspects of Enterprise Workflow. You will use the administrative interface to maintain users, user groups, access lists (security), workbasket and workpiece classes. It can also be used to search for active workpieces, analyze workpiece history, advance a workpiece or stop a workpiece.

When you log into Enterprise Workflow using the administrative interface you will be presented with a screen similar to the one below.

If a user logs into workflow using the administrative client, and the user does not have authority to most features, such as the ability manage or maintain users, then that user will not be allowed to complete those unauthorized tasks.

Refer to the following sections in this manual for help on each option within the administrative interface.
Create Workpiece

Although most Enterprise Workflow users will either use our Workbasket user interface or their own custom interface to enter and advance workpieces through the system you can create new workpiece items from the Administration page. In general, you will probably only use this portion of the administration interface during test phases or special one-off circumstances to create a new workpiece.

A sample of the initial workpiece screen is shown below.

Each new workpiece must be associated with a workpiece class and process. The Workpiece class will determine the type of information that will be maintained for the workpiece through the defined attributes of the workpiece. The Process will determine which business process the workpiece item will follow.

Once you have selected the class and process you can begin to associate documents from your document repository, such as RJS WebDocs document management system, to the workpiece. You may reference as many documents as necessary for the particular workpiece. Each document reference is created by entering a document URL, link to the specific document, a description of the document and a document type. If you were creating a workpiece for the purchase order process, you might reference a vendor invoice. When you do so, the description might say “Vendor ABC Company” and the Type “Invoice”.

Workpiece attributes are used to enter index key values for the workpiece. These indexes are defined by the workpiece class as Column Names. When entering the column name into the Name edit box, you must enter the Column Name exactly as it listed in the class. Column Names are case-sensitive. You may enter one value for each column name available for the workpiece class. Again, if you are creating a workpiece for a purchase order process, then you might have column names such as VendorNumber and PONumber.
Workflow Engine Status

Maintain Workbaskets

Remember, workbaskets are similar in nature to your inbox. Workbaskets act as holding tanks for items of work. All workbaskets for a process must be created prior to defining your business processes in Workflow Designer.

You may choose to create general workbaskets that can be accessed by multiple users or you may decide to create separate workbaskets for each user within the process. The decision as to how you create your workbaskets will be based on your analysis and design of the business process.

When you select Maintain Workbaskets from the navigation panel on the left side of the interface, you will be presented with a screen similar to the one shown below.

![Maintain Workbaskets - Initial Screen](image)

Workbasket Fields

*Workbasket Name:* Use this field to enter a descriptive name for the workbasket. Workbasket names might represent the name of the step they represent, e.g., Purchase Orders, or they may represent the name of the person to process the workpieces at a particular step.

*Access List:* Select an access list which Enterprise Workflow engine will use to secure this workbasket and its contents during processing. Refer to the Maintain Access List section for detailed help on the use of Access Lists.

*Forward:* By enabling this option, documents associated with workpieces placed in this workbasket will be forwarded to the specified email address. Enter the email address of the person to forward to in the box to the left of the option check box.

*Notify:* By enabling this option, the specified person will be notified via email each time a workpiece is placed in this workbasket. Enter the email address of the person to notify when items are placed in this workbasket.
Maintain Workpiece Class

Workpiece Class is used to define the type of reference information, also known as workpiece attributes, to be collected for each workpiece and to select the Access List to be used to secure the workpieces.

Each Workpiece Class can have one to many columns associated with it. These columns will be used to store attributes associated with the workpiece class. For example, a purchase order class might maintain attributes values for Vendor #, Purchase Order #, Invoice #, Invoice Date, Purchase Order Date, etc. Attributes will be used both as necessary reference information and for search purposes.

When you select Maintain Workpiece Class from the navigation panel on the left side of the interface, you will be presented with a screen similar to the one shown below.

Workpiece Class Fields

**Class Name:** Use this field to enter a descriptive name for the workpiece class. Workpiece classes are used to categorize the various types of workpiece items that are being maintained by Enterprise Workflow. Common workpiece classes would be AP Invoice, Purchase Orders, Resume, or Patient Information.

**Access List:** Select an access list which Enterprise Workflow engine will use to secure this workpiece class during processing. Refer to the Maintain Access List section for detailed help on the use of Access Lists.

**Description:** Use this field to provide a more descriptive name for the workpiece class than is used as the Class Name.

**Column List:** Enter as many column names, also considered attributes, as is necessary for the workpiece class. These attributes represent the key indexes you need to maintain for the workpiece class. Users will be able to enter values associated with each of these attributes for every workpiece. These values can also be used during the processing of the workpiece to
determine how the workpiece proceeds through the process. For example, in a purchase order process you may collect the PO Amount and then use that amount at a decision point to determine which workbasket to place the workpiece item. The first three column names included in the list will be displayed on pages such as the Active Workpieces to help you identify the workpiece.

**Maintain Access List**

Access Lists are used to grant/restrict authority of various assets within Enterprise Workflow. Access lists can be associated with users, workbaskets and workpiece classes. You may use the same access lists to control similar assets; for example, you may create an access list for accounts payable (AP) and then assign the access list to AP users, workbaskets and workpiece classes. In contrast, you may choose to create access lists that are based on functionality or tasks such as one access list for supervisors and another for standard data entry clerks.

When you select Maintain Access List from the navigation panel on the left side of the interface you will be presented with a screen similar to the one shown below.

**Access Lists Fields**

**Access List Name**: Use this field to enter a name for the Access List.

**Description**: Use this field to enter a longer descriptive name for the Access List. For example, you may choose to create an access list with the name of Admin and then use Administrators as your description.

**Users**: This list box displays either a list of all groups or individual users currently defined for Enterprise Workflow. The selection of group or User determines which values are listed in the list box.

**Unassigned privileges**: This list box displays a list of all Enterprise Workflow privileges that are currently not associated with the selected user/group. Please note, both the Unassigned privileges and the Privileges allowed for user will be blank if you do not highlight (select) a user/
Privileges allowed for user: This list box displays a list of all Enterprise Workflow privileges that have been assigned to the selected user/group. Please note, both the Unassigned privileges and the Privileges allowed for user will be blank if you do not highlight (select) a user/group from the Users list. Use the right and left arrows to move privileges to/from the Unassigned and Allowed lists.

Available Privileges

**addWorkpieceItem**: Used to associate a new reference item, such as a scanned invoice or document, with a workpiece. Typically, these documents will be managed and stored in an electronic repository; RJS WebDocs is an excellent example of an online document repository.

**addWorkpieceNote**: Used to insert new Note that will be associated with the current workpiece.

**addWorkpieceValue**: Used to insert attribute values the workpiece; for example, add Social Security Number to a workpiece.

**advanceWorkpiece**: Used to move a workpiece from the current step in the workflow process to the next step.

**createAccessList**: Used to define a new Access List.

**createClass**: Used to define a new Workpiece Class.

**createGroup**: Used to define a new User Group.

**createUser**: Used to define a new workflow user profile.

**createWorkbasket**: Used to define a new workbasket.

**createWorkpiece**: Used to insert new workpiece into a workflow process. This can be accomplished through the Create Workpiece option in either Admin or Workbasket interfaces or programmatically.

**deleteClass**: Used to permanently remove a workpiece class from the Enterprise Workflow environment.

**deleteWorkbasket**: Used to permanently remove a workbasket from the Enterprise Workflow environment. Since this function can have serious repercussions with your Enterprise Workflow environment we strongly suggest that you limit user authority to this command.

**getAccessListDetail**: This function is used to retrieve all particulars regarding privileges assigned to users and/or groups for a specific Access List. So if you selected the Admin access list from the maintenance screen you would be able to review the specific privileges that are associated with the users/groups for the Admin access list.

**getAccessListInfo**: This function is used to retrieve the general information associated with an access list: description, creation time stamp and user that created the access list.

**getAccessListList**: Used to retrieve a list of all defined access lists in Enterprise Workflow.

**getActiveWorkpieceList**: Used to retrieve a list of all active, unfinished, workpieces throughout all processes within Enterprise Workflow.
**getClassList**: Used to retrieve a complete list of workpiece classes.

**getPrivilegeList**: This command retrieves a current list of available privileges that can be used to define workflow security.

**getProcessList**: Used to retrieve a list of processes currently defined, or at least in the process of being defined, for Enterprise Workflow. When you are using Workflow Designer and choose Retrieve from the Process menu, this function is used to return the list of processes.

**getSteps**: Used to retrieve the process steps defined for the specified workflow process.

**getUserGroupList**: Retrieves a list of all defined user groups in Enterprise Workflow.

**getUserGroupMemberList**: Retrieves a list of all users that are currently associated with the specified user group.

**getUserList**: Used to retrieve a list of all user profiles currently defined in Enterprise Workflow.

**getWorkbasketList**: This function returns a list of all workbaskets defined in Enterprise Workflow.

**getWorkbasketTOC**: Each workpiece within a workbasket has indexes or columns of information associated with it as defined in the workpiece class. The first three columns in the workpiece class (the top three items shown in a columns list) are considered the Table of Contents (TOC) for the workbasket. This function returns the TOC values for the specified workbasket.

**getWorkpieceClassColumnList**: Used to retrieve a list of the defined columns for the specified workpiece class.

**getWorkpieceHistory**: This function returns a history log for the specified workpiece. The history log includes information such as date of activity, user, From step ID number and name, event that occurred (such as “moved by normal processing”), To step ID number and name, and the workpiece priority.

**getWorkpieceInfo**: This function is used to retrieve all relevant links and index values currently associated with the specified workpiece. For example, if a user opened up a workpiece to either enter or check information entered for the workpiece, the data displayed on the screen is the information returned when executing the method getWorkpieceInfo.

**getWorkpieceListByValues**: This function returns the current list of values entered for the specified workpiece.

**getWorkpieceNotes**: Used to retrieve all of the notes currently linked to the specified Workpiece.

**makeProcess**: This function is used by Workflow Designer when a user chooses to save a new process definition.

**moveWorkpiece**: This function is used whenever a workpiece is moved, either through normal processing of a user or as a special move by an administrator. The move may be within the current workflow or possibly to a different workflow process.

**removeAccessList**: Used to delete the specified access list. Be cautious when using this function. If you remove an access list that is linked to assets such as users, workbaskets or workpiece classes, you take the risk of creating a security gap in your workflow assets.

**removeGroup**: Used to delete the specified user group. If you are using user groups to simplify
your workflow security, removing a user group can eliminate all security, associated with all users within the group.

**removeUser**: This function will delete the specified user from Enterprise Workflow.

**removeWorkpieceItem**: The specified workpiece item, typically a document reference to an external document repository, will be removed from the workpiece.

**removeWorkpieceValue**: Used to remove an entered value for the workpiece. For example, if you have a column/name for SSN and the SSN is incorrect for the workpiece, then this function can be used to remove the erroneous value.

**replaceAccessListDetail**: This function is used whenever an access list is updated. As an example, if you add additional approved functions to the access list, then the replaceAccessListDetail function will be used.

**replaceUserGroupMemberList**: Used to update the included members within the specified user group. Included members can be added or removed from the list.

**setWorkpieceClassColumnList**: Used to add/change/delete columns associated with a workpiece class.

**setWorkpiecePriority**: This function is used whenever a user requests to change the priority of a workpiece item.

**stopWorkpiece**: Used to end the workpiece processing. The halting of processing of a workpiece can be due to a special request by an authorized user or because the workpiece has reached the end of the defined workflow process.

**updateClass**: Used whenever the details of a workpiece class are updated.

**updateWorkbasket**: This function is called when the details of a workbasket, such as to change the access list associated with the workbasket, are changed.

### Maintain User Account

A user account, or profile, must be created for each user that will be using Enterprise Workflow. User names and passwords are case sensitive.

When you select Maintain User Account from the navigation panel on the left side of the interface you will be presented with a screen similar to the one shown below.
User Account Fields

**User Name**: Use this field to enter a user ID or name for the user. This field is case sensitive. If the user name is entered using proper case, then the user will be required to use proper case each time he or she logs into Enterprise Workflow.

**Description**: Enter a descriptive name for the user. Typically this field will be used to enter the user's full first and last name so that the user is clearly identified.

**Access List**: Select the desired access list for the user. Security restrictions associated with the Access List assigned to the user account will take precedence over any other access list. In other words, an access list associated with a workbasket cannot give more authority to the user profile than is already associated with the access list selected here on the user account.

**Password**: Use this field to enter a pass code associated with the user profile. Passwords are case sensitive. If the password is entered in all capital letters, then the user must always enter the password in upper case.

Maintain User Group

User Groups allow you to assign security privileges by groups of users rather than individually. Create user groups based on required functions within Enterprise Workflow. Then define an access list that can be applied to the user group. By doing this, you avoid the need to individually assign and manage privileges to each user.

You are not required to use User Groups but as mentioned above they can reduce management of workflow security.

When you select Maintain User Group from the navigation panel on the left side of the interface you will be presented with a screen similar to the one shown below.
User Group Fields

**User Group Name**: Use this field to enter a name for the user group. User Group names may represent a particular group of users, such as Accounts Payable, or to represent a security level within Enterprise Workflow.

**Description**: Enter a descriptive name for the User Group. For example, you may use the shorter text of Admin as the User Group name and then enter the complete word Administrators as the description.

**Members**: Displays a list of members, workflow users, which are currently associated with the user group. By selecting users from the list of Non-Members on the right and then clicking the right arrow you include the user as a member to the group. When you click on the right arrow, the selected user name will be moved from the Non-Members list to the Members list.

**Non-Members**: Displays a list of workflow users that currently are not considered members of the User Group. By selecting users from the list of Members on the left and then clicking the left arrow you can remove a user from the group. When you click on the left arrow the selected user name will be moved from the Members list to the Non-Members list.

Workpiece Status

Search for Workpiece

Use the Search for Workpiece to find workpiece, active or inactive. The initial search screen is shown below.
To use the Search screen enter the workpiece search criteria. You must enter at least one criterion for the search. If you fail to enter any criteria and then select to search, the results screen will be empty. If you enter multiple search criteria, only those workpieces that satisfy all criteria, in other words AND comparisons, will be returned in the results list.

Each criterion is entered by typing the column name, as defined in the workpiece class, and column name value your are interested in searching for and then click Add to Search button. You must click on the button Add to Search to register the criterion for the search. Continue to add your desired criterion until all have been entered. When entering Column Name, you must type the name exactly as it has been defined in the Workpiece Class. The column name is case-sensitive. Below is a sample results list.
Active Workpiece List

The Active Workpiece list includes a listing of all workpieces that are currently being processed within any workflow process. You will not be given any opportunity to search for a subset of the active workpieces. All Active workpieces are included. Any workpiece that has reached the end of a business process, the Stop point, will not be included in this list.

A sample list of active workpieces is shown below.
You may look at the detailed history associated with any workpiece by selecting the workpiece from the results list. The detail screen will include the same general information that was shown in the results list and then all of the activity history for the workpiece. The detail will show you date/time stamp of the activity, user, initial workflow step ID and description, event description, to step ID and description and workpiece priority. The history detail can be useful in tracing why a workpiece is at a particular step or who processed the item at any given point in the process. You can also reach this same history information from the Workpiece History option on the left task bar.
Workpiece History

The Workpiece History provides you with a method to view all of the activity associated with one workpiece. The history will show you who has worked on an item, when it was worked on, and where it moved from there. This is an excellent audit log for any particular workpiece. You must know the workpiece ID to find the item.

To look at the history of a workpiece, enter the Workpiece ID and click submit. Below is a sample of the workpiece history that will be displayed for a workpiece.
Move Workpiece

If you have a workpiece that needs to be moved from one workbasket to another, you can use the Move Workpiece option. To move a workpiece you will need to know the workpiece ID number, the new step ID, the workbasket to move the workpiece to, and the event string that will be used to help define the manual move. Your user profile must have the been assigned the moveWorkpiece authority to complete this task.

Below shows a sample of how you might complete the values for the Move Workpiece.

After the move has been submitted you can use the Workpiece History option to view the move.
Stop Workpiece

Use the Stop Workpiece option to manual stop, or remove, a workpiece from a process flow. Once stopped, this workpiece will no longer be included in the active workpieces for the process flow. A workpiece can be stopped at any time in the process. Your user profile must have been assigned the stopWorkpiece authority to complete this task.

To stop a workpiece you must know the workpiece ID, process step ID and the event information. A sample screen with the necessary information is shown below.
Once a workpiece has stopped the detail will look similar the detail shown below. Notice, the last action listed for the workpiece is the manual stop action performed by Sherri at the shown time.
Chapter 7

Integrating with WebDocs for iSeries
Integrating with WebDocs for iSeries

A certain amount of customization of your WebDocs installation must be done to integrate it with RJS Enterprise Workflow.

In general, you choose which document types you will use as the basis for creating workpieces. You'll configure the document types according to the documentation for WebDocs. You'll then create a new DOCEXITC program which is called by the DOCCHKIN command. DOCCHKIN is called by every application that adds documents to WebDocs. For instance, the RJS WebDocs Scan Workstation calls this command. The WebDocs web interface also calls this command when adding documents to WebDocs. Therefore, any document added to WebDocs will pass through the DOCCHKIN command which will call your DOCEXITC program.

You should use the MAINURL data area to set up the base URL for your WebDocs HTTP server. Then use the contents of this data area in your DOCEXITC program to set up the URL for each WebDocs document reference set up in a workflow workpiece.

**MAINURL *dtaara**

RJSIMAGE library should contain a data area with the URL of the Image Server. The contents of the data area should be something like this. The host portion should reflect the DNS name or IP address of the AS400 machine or partition where the RJS WebDocs Image Server runs. The port should reflect the port monitored by the web server.

Display Data Area

<table>
<thead>
<tr>
<th>System:</th>
</tr>
</thead>
<tbody>
<tr>
<td>S105712M</td>
</tr>
<tr>
<td>Data area . . . . . . . . . . :</td>
</tr>
<tr>
<td>Library . . . . . . . . . . :</td>
</tr>
<tr>
<td>Type . . . . . . . . . . :</td>
</tr>
<tr>
<td>Length . . . . . . . . . :</td>
</tr>
<tr>
<td>Text . . . . . . . . . :</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Offset</td>
</tr>
<tr>
<td>0</td>
</tr>
<tr>
<td>50</td>
</tr>
<tr>
<td>100</td>
</tr>
<tr>
<td>150</td>
</tr>
</tbody>
</table>
DOCCHKIN Exit program DOCEXITC

The following is an example of DOCEXITC which can be used to tightly integrate Enterprise Workflow with RJS WebDocs document management system. Contact RJS support personnel for additional help with product integration and automation.

```c
/*
   Author: Vernon M. Hamberg, RJS Software Systems
   Date written: 06/30/2005
   Purpose: Post-checkin exit point
   Description: Run commands to create workpiece
*/
PGM PARM(&DOCID &REVISION &DOCTITLE &DOCFLR1 &DOCFLR2 &DOCFLR3 +
   &KEY1 &KEY2 &KEY3 &KEY4 &KEY5 &KEY6 &KEY7 &KEY8 &KEY9 +
   &KEY10 &DOCPATH &DOCFILE &DOCTYPE &DOCTYPE2)
DCL VAR(&DOCID) +
   TYPE(*CHAR) +
   LEN(100)
DCL VAR(&REVISION) +
   TYPE(*DEC) +
   LEN(9 0)
DCL VAR(&DOCTITLE) +
   TYPE(*CHAR) +
```
LEN(200)

DCL VAR(&DOCFLR1) +
    TYPE(*CHAR) +
    LEN(100)

DCL VAR(&DOCFLR2) +
    TYPE(*CHAR) +
    LEN(100)

DCL VAR(&DOCFLR3) +
    TYPE(*CHAR) +
    LEN(100)

DCL VAR(&KEY1) +
    TYPE(*CHAR) +
    LEN(200)

DCL VAR(&KEY2) +
    TYPE(*CHAR) +
    LEN(200)

DCL VAR(&KEY3) +
    TYPE(*CHAR) +
    LEN(200)

DCL VAR(&KEY4) +
    TYPE(*CHAR) +
    LEN(200)

DCL VAR(&KEY5) +
    TYPE(*CHAR) +
    LEN(200)

DCL VAR(&KEY6) +
    TYPE(*CHAR) +
    LEN(200)

DCL VAR(&KEY7) +
    TYPE(*CHAR) +
LEN(200)
DCL  VAR(&KEY8) +
     TYPE(*CHAR) +
     LEN(200)
DCL  VAR(&KEY9) +
     TYPE(*CHAR) +
     LEN(200)
DCL  VAR(&KEY10) +
     TYPE(*CHAR) +
     LEN(200)
DCL  VAR(&DOCPATH) +
     TYPE(*CHAR) +
     LEN(255)
DCL  VAR(&DOCFILE) +
     TYPE(*CHAR) +
     LEN(255)
DCL  VAR(&DOCTYPE) +
     TYPE(*CHAR) +
     LEN(10)
DCL  VAR(&DOCTYPE2) +
     TYPE(*CHAR) +
     LEN(20)
DCL  VAR(&BASEURL) +
     TYPE(*CHAR) +
     LEN(256)

**************************************************************************
/* Library where Workpiece Creation Commands live */
**************************************************************************
DCL  VAR(&CRTWKPLIB) +
     TYPE(*CHAR) +
LEN(10) +
VALUE('RJSFLOW')
/*****************/
/* Logical constants */
/* So we'll know when to RMVLIBLE */
/*******************/
DCL VAR(&TRUE) +
    TYPE(*CHAR) +
    LEN(1) +
    VALUE('1')
DCL VAR(&FALSE) +
    TYPE(*CHAR) +
    LEN(1) +
    VALUE('0')
DCL VAR(&CMDSINLIBL) +
    TYPE(*CHAR) +
    LEN(1)
/*******************/
/* Workpiece variables */
/*******************/
DCL VAR(&UNIQUEID) +
    TYPE(*CHAR) +
    LEN(8)
DCL VAR(&URL) +
    TYPE(*CHAR) +
    LEN(1800)
DCL VAR(&CAPTION) +
    TYPE(*CHAR) +
    LEN(200)
/*******************/
/* Error handler */

/*****************/

DCL VAR(&ERROR) +

   TYPE(*CHAR) +
   LEN(1) +
   VALUE('0')

DCL VAR(&MSGKEY) +

   TYPE(*CHAR) +
   LEN(4)

DCL VAR(&MSGTYP) +

   TYPE(*CHAR) +
   LEN(10) +
   VALUE('*DIAG')

DCL VAR(&NBRTYP) +

   TYPE(*CHAR) +
   LEN(4) +
   VALUE(x'00000001')

DCL VAR(&PGMMSGQ) +

   TYPE(*CHAR) +
   LEN(10) +
   VALUE('**')

DCL VAR(&STKCTR) +

   TYPE(*CHAR) +
   LEN(4) +
   VALUE(x'00000001')

DCL VAR(&ERRCODE) +

   TYPE(*CHAR) +
   LEN(4) +
   VALUE(x'00000000')

/* Global error handler */
MONMSG    MSGID(CPF0000) +
         EXEC(GOTO CMDLBL(ERRORS))
/**-----------------------------*/
/* Add libraries to library */
/* list - add more as needed */
/* for access to your data */
/**-----------------------------*/
ADDLIBLE   LIB(&CRTWKPLIB)
MONMSG    MSGID(CPF2103) +
         EXEC(DO)
         CHGVAR   VAR(&CMDSINLIBL) +
         VALUE(&TRUE)
ENDDO
/**-----------------------------*/
/* Is it a doctype to be processed? */
/**-----------------------------*/
IF        COND(&DOCTYPE2 = 'CLAIMS') +
         THEN(DO)
/**-----------------------------*/
/* Call procedures to create workpiece */
/* */
/* GETUNQID starts a new workpiece */
/* CRTWKPVVAL adds a value */
/* (add a many as you like) */
/* CRTWKPREF adds this doc to it */
/* (could add more if ???) */
/* CRTWRKPCF finishes the creation */
/**-----------------------------*/
RTVDATAARA   DTAARA(MAINURL) RTNVAR(&BASEURL)
CRTWKPVVAL   UNIQUEID(&UNIQUEID) NAME('Document Type') +
VALUE(&KEY6)
CRTWKPVAL UNIQUEID(&UNIQUEID) NAME('SSN') +
    VALUE(&KEY1)
CRTWKREFIX UNIQUEID(&UNIQUEID) URL(&BASEURL *TCAT '/' +
            *TCAT +
            'IMAGESERVER/DOC100R?ACTION=VIEW&IDOCID=' +
            *TCAT &IDOCID) CAPTION(&DOCTYPE)
CRTCWRKPKCE UNIQUEID(&UNIQUEID) CLASS('Claims') +
    PROCESS('Test Basic Flow') +
    DESCRIP(&DOCTYPE)
/**.................................**/
/* That's all for CLAIMS */
/**.................................**/
ENDDO
/**.................................**/
/* Reset library list to what */
/* what it was before this */
/* was called */
/**.................................**/
IF COND(*NOT (&CMDSINLIBL *EQ &TRUE)) +
    THEN(DO)
RMVLIBLE LIB(&CRTCWRKPLIB)
MONMSG MSGID(CPF2104)
ENDDO
RETURN /* NORMAL EXIT */
/**.................................**/
/* General error handler */
/**.................................**/
ERRORS:
/* Prevent error-handling loop */
IF COND(&ERROR *EQ &TRUE) +

THEN (DO)

GOTO CMDLBL(ENDPGM)

ENDDO

ELSE CMD (DO)

CHGVAR VAR(&ERROR) +

VALUE (&TRUE)

ENDDO

*******************************************************************************
/* Reset library list to */
/* what it was before this */
/* was called */
*******************************************************************************

IF COND (*NOT (&CMDSINLIBL *EQ &TRUE)) +

THEN (DO)

RMVLIBLE LIB (&CRTWKPLIB)

MONMSG MSGID (CPF2104)

ENDDO

/* Move all *DIAG messages to previous program queue */

CALL PGM (QMHHMOVPM) +

PARM (&MSGKEY &MSGTYP &NBRTYP &PGMMMSGQ &STKCTR &ERRCODE)

/* Resend last *ESCAPE message */

CALL PGM (QMHRSNEM) +

PARM (&MSGKEY &ERRCODE)

ENDPGM:

ENDPGM
Chapter 8

Web Service API
Web Service API

As noted earlier in this manual, Enterprise Workflow is completely customizable and can be integrated with any of your business processes through the use of the API methods (in essence, web service commands). With these APIs you have the ability to create your own applications and/or interfaces to use with the Enterprise Workflow engine. Or, you may choose to use the APIs for specific integration points with your current business processes and still allow your users to use the provided Enterprise Workflow interfaces.

This chapter of the book is very technical in nature and is intended as a programmer’s reference to using the Enterprise Workflow APIs. Each API description includes the required syntax, description of API parameters, expected return values and what to expect if an error occurs when executing the given API.

General Processing Actions

All of the methods described in this section are associated with basic processing type actions (common user tasks) completed through Enterprise Workflow. In general, these methods will allow you to create, change or delete workpiece details such as notes, references and attribute values.

Workpiece Maintenance

This section describes those methods associated with the creation and maintenance of Enterprise Workflow workpieces.

createWorkpiece

Description

Use to generate a new workpiece item; column name values and references can be included during the creation process.

Syntax

createWorkpiece(java.lang.Object[] object)

Parameters:

object - a deep array object[] that contains workpiece details. Object[] is comprised of three elements, each of which is an array of values.

Element 1 - is an array of workpiece attributes.

- Class - represents the associated workpiece class for this workpiece.
- ProcessID - represents the desired process ID to associate with the workpiece.
- Priority - represents the processing priority for the new workpiece.
- Description - represents text description to help identify the new workpiece.

Element 2 - is an array of column names and associated values.
• Value Name - represents the column name (as defined in the workpiece class) to associate the value with.

• value value - represents the actual value linked to the column name.

Element 3 - is an array of item reference details.

• URL - represents the URL address to be used to access the referenced document image.

• Type - represents the document type of the referenced document image (e.g., DOC or JPEG).

<table>
<thead>
<tr>
<th>Class</th>
<th>ProcessID</th>
<th>Priority</th>
<th>Description</th>
<th>Value Name 1</th>
<th>Value Value 1</th>
<th>URL 1</th>
<th>Type 1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Value Name 2</td>
<td>Value Value 2</td>
<td>URL 2</td>
<td>Type 2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Value Name n</td>
<td>Value Value n</td>
<td>URL x</td>
<td>Type x</td>
</tr>
</tbody>
</table>

createWorkpiece Array Structure

Returns:

0 if successful; -1 otherwise

Throws:

java.lang.Exception

getWorkpieceInfo

Description

Use to retrieve an array listing of all the details currently entered for a particular workpiece.

Syntax

getWorkpieceInfo(java.lang.String workpieceID)

Parameters:

workpieceID - a string value that represents the workpiece ID that you wish to retrieve the particulars on.

Returns:

Will return a deep array Object[] that contains workpiece information. Object[] is comprised of four elements, each of which is an arrays.

Element 1 - is an array of the workpiece table of contents (TOC).
- **Step ID** - represents the numeric value of the current step in the process.
- **Option ID** - represents the numeric value of the current option within the step.
- **Option Value** - represents the actual value of the associated option ID for the step. For example, the Option Value may contain a Boolean string comparison.

*Element 2* is an array document reference details.

- **Item ID** - represents the numeric item identification for the document reference.
- **URL** - represents the actual URL address to be used to access the referenced document.
- **Type** - represents the type of document to be referenced (e.g., DOC or JPEG).
- **Created By** - represents the workflow user that created/entered the document link.
- **Date** - a date/time stamp that represents the date and time that the document link was created or last changed.

*Element 3* is an array of currently entered attributes and the associated values.

- **Value Name** - represents the column name (defined in the Workpiece Class).
- **Value Value** - represents the actual value associated with the given column name.

*Element 4* is an array of general workpiece details.

- **Workpiece ID** - represents the unique numeric workpiece ID that is used to identify the specific workpiece.
- **Class** - represents the associated workpiece class for this workpiece.
- **Process ID** - represents the desired process ID to associate with the workpiece.
- **Priority** - represents the processing priority for the new workpiece.

<table>
<thead>
<tr>
<th>(Element 1 - 0 to x elements)</th>
<th>(Element 2 - 0 to n elements)</th>
<th>(Element 1 - 0 to x elements)</th>
<th>(Element 2 - 0 to n elements)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step ID 1</td>
<td>Option ID 1</td>
<td>Option Value 1</td>
<td>Item ID 1</td>
</tr>
<tr>
<td>Step ID 2</td>
<td>Option ID 2</td>
<td>Option Value 2</td>
<td>Item ID 2</td>
</tr>
<tr>
<td>Step ID x</td>
<td>Option ID x</td>
<td>Option Value x</td>
<td>Item ID y</td>
</tr>
</tbody>
</table>

*Left Side of getWorkpieceInfo Array Structure*
Right Side of `getWorkpieceInfo` Array structure

Throws:

java.lang.Exception

**addWorkpieceItem**

Description

Use to generate a new workpiece item; column name values and references can be included during the creation process.

Syntax

```
addWorkpieceItem(java.lang.String workpieceID, java.lang.Object[] item)
```

Parameters:

- **workpieceID** - a string identifying the workpiece.
- **item** - a two-element string array that contains document image reference details.
  - URL - represents the actual URL address to be used to access the referenced document.
  - Type - represents the type of document to be referenced (e.g., DOC or JPEG).

Returns:

0 if successful

Throws:

java.lang.Exception

**getWorkpieceNotes**

Description

Use this method to retrieve the entire string of workpiece notes.

Syntax
**getWorkpieceNotes** (java.lang.String workpieceID)

**Parameters:**

- workpieceID - a string parameter that represents the workpiece ID that notes will be retrieved for.

**Returns:**

- The complete note string. In the interface notes appear as multiple values, in actuality they are stored as one long variable-length string.

**Throws:**

- java.lang.Exception

**addWorkpieceNote**

**Description**

This method adds the string to the current notes object along with a header that includes who added the note and a date/time stamp of when the note was added.

**Syntax**

```java
addWorkpieceNote(java.lang.String workpieceID, java.lang.String note)
```

**Parameters:**

- workpieceID - a string value that represents the workpiece ID that the new note is to be linked to.
- note - a string value that represents the new note and audit who/when header.

**Returns:**

- 0 if successful; -1 otherwise

**Throws:**

- java.lang.Exception

**addWorkpieceValue**

**Description**

Use this method to add a name/value pair to a workpiece.

**Syntax**

```java
addWorkpieceValue(java.lang.String workpieceID, java.lang.Object[] item)
```

**Parameters:**

- workpieceID - the workpiece to which to add the name and value to.
item - Object[] array with name as the first element and value as the second.

Returns:
0 if successful; -1 otherwise

Throws:
java.lang.Exception

**removeWorkpieceItem**

Description
This method can be used to remove an item from a workpiece TOC.

Syntax
```
removeWorkpieceItem(java.lang.String workpieceID, java.lang.String itemID)
```

Parameters:
- workpieceID - a string value that represents the workpiece from which to remove the item.
- itemID - a string value that represents the item to be removed.

Returns:
0 if successful; -1 otherwise

Throws:
java.lang.Exception

**removeWorkpieceValue**

Description
Use this method to remove a name/value pair from a workpiece.

Syntax
```
removeWorkpieceValue(java.lang.String workpieceID, java.lang.String valueName)
```

Parameters:
- workpieceID - a string value that represents the workpiece from which to remove the name/value.
- valueName - a string value that represents the name of the name/value to remove.

Returns:
0 if successful; -1 otherwise

Throws:
moveWorkpiece

Description

Use this method to move the workpiece to a different workbasket without changing the step.

Syntax

moveWorkpieceValue(workpieceID, stepID, workbasket, eventInfo)

Parameters:

- workpieceID: a string value that represents the workpiece from which to remove the name/value.
- stepID: a string value that represents the step ID number where the workpiece currently is. If the workpiece is not at this step when the function is executed, the workpiece will not be moved and no exception will be thrown.
- workbasket: a string value that represents the name of the workbasket to which the workpiece should be moved.
- eventInfo: a string value that represents the text string copied to the workpiece history.

Returns:

0 if successful; -1 otherwise

Throws:

java.lang.Exception

stopWorkpiece

Description

This method moves the workpiece to a STOP step.

Syntax

stopWorkpiece(workpieceID, stepId, eventInfo)

Parameters:

- workpieceID: a string value that represents the workpiece ID that will be moved to the stop step.
- stepId: a string value that represents the step ID number where the workpiece currently is. If the workpiece is not at this step when the function is executed, the workpiece will not be stopped and no exception will be thrown.
- eventInfo: a string value that represents a text description as to why the workpiece was
moved to the stop step.

**Throws:**

java.lang.Exception

### advanceWorkpiece

**Description**

Use this method to move a workpiece to the next step on the workflow process. The workpiece will continue to advance one step at a time until it reaches a "stopping" point on the workflow process. Stopping points include Work Points and Collection Points.

**Syntax**

```java
advanceWorkpiece(java.lang.String workpieceID, java.lang.Object[] option)
```

**Parameters:**

- **workpieceID** - a numeric value that represents the system workpiece ID that will be advanced.
- **option[]** - an array of string values that represent a chosen option. The array includes the step ID, the option ID and the option value. For example, in a purchase order workflow the option may be "approved" which moves the workpiece to the end of the process. Look through the WorkpieceInfo array for the option array that matches the "approved" value and pass that array. If the workpiece is no longer at the step, the function will do nothing and will throw no exception.

  - **Step ID** - represents the numeric value of current step in the process.
  - **Option ID** - represents the numeric value of the current option within the step.
  - **Option Value** - represents the actual value of the associated option ID for the step. For example, the Option Value may contain a Boolean string comparison.

**Returns:**

0 if successful; -1 otherwise

**Throws:**

java.lang.Exception

### getWorkpieceHistory

**Description**

Use this method to retrieve a list of all the history records for the workpiece. Workpiece history consists of one record for each step through the process. The output array is composed like the database table.

**Syntax**

```java
getWorkpieceHistory(java.lang.String workpieceID)
```
Parameters:

workpieceID - a string value that represents the workpiece ID to retrieve history for.

Returns:

An array of workpiece history entries.

- Date - a date/time stamp that represents the date and time that the document link was created or last changed.
- User - the name of the user that completed the step.
- From Step ID - contains either the unique step ID number, such as 15, or values such as START or END associated with the step the workpiece has moved from.
- From Step Type - contains a description of the step such as START, WORKBASKET, or END associated with the step the workpiece has moved from.
- Event String - contains a text description of what happened during the step.
- To Step ID - contains either the unique step ID number, such as 15, or values such as START or END associated with the step the workpiece moved to.
- To Step Type - contains a description of the step such as START, WORKBASKET, or END associated with the step the workpiece moved to.
- To Step Name - contains a text description of the step in which the workpiece has moved.
- Priority - represents the processing priority for the new workpiece.

Throws:

java.lang.Exception

Workpiece Lists

This section describes those methods associated with the retrieval of lists related to a workpiece.

getWorkbasketTOC

Description

Use this method to retrieve an array listing all the workpieces awaiting attention in the named workbasket.

Syntax

getWorkbasketTOC(java.lang.String workbasket)

Parameters:

workbasket - a string value that represents the name of the workbasket to retrieve active workpiece items from.
Returns:

Object[] array of workpiece information; each element of the array is an array of workpiece attributes: Workpiece ID, Class, Process ID, Description, Priority and the values of the first three attributes named in the column list associated with the workpiece class.

- Workpiece ID - a string value that uniquely represents desired workpiece.
- Class - represents the associated workpiece class for this workpiece.
- ProcessID - represents the desired process ID to associate with the workpiece.
- Description - represents text description to help identify the new workpiece.
- Priority - represents the processing priority for the new workpiece.
- Column 1 - represents the value associated with the first column name (attribute) as defined in the workpiece class.
- Column 2 - represents the value associated with the second column name (attribute) as defined in the workpiece class.
- Column 3 - represents the value associated with the second column name (attribute) as defined in the workpiece class.

Throws:

java.lang.Exception

getWorkpieceListByValues

Description

This method returns a list of all the workpieces having the specified key(s) and value(s) associated to it.

Syntax

getWorkpieceListByValues(java.lang.Object[] keyValuePairs)

Parameters:

keyValuePairs - a two element string array with the key name in the first string and the value of that attribute in the second string.

Returns:

An Object[] array of workpiece information; each element of the array being an array of workpiece details.

- Workpiece ID - a string value that uniquely represents desired workpiece.
- Attribute Name - represents the column name (as defined in the workpiece class) to associate the value with.
- Attribute Value - represents the actual value linked to the column name.
• Type - represents the type of value (always String at this writing).

Throws:

java.lang.Exception

**getActiveWorkpieceList**

**Description**

This method returns an array of all the active workpieces, that is, all the workpieces whose current step is not a STOP step.

**Syntax**

`getActiveWorkpieceList()`

**Parameters:**

No parameter associated with this method.

**Returns:**

An Object[] array of active workpiece information.

• Workpiece ID - a string value that uniquely represents desired workpiece.

• Class - represents the associated workpiece class for this workpiece.

• Created Date - a date/time stamp that represents the time and date the workpiece was created.

• Created By - represents the user that created the workpiece.

• Current Step Type - represents the type of step the workpiece is at. For example, a collection point or a workbasket.

• Current Step Name - a string value that represents the name of the step as defined in the workflow process.

• Priority - represents the processing priority for the new workpiece.

• Description - represents text description to help identify the new workpiece.

• Column 1 - represents the value associated with the first column name (attribute) as defined in the workpiece class.

• Column 2 - represents the value associated with the second column name (attribute) as defined in the workpiece class.

• Column 3 - represents the value associated with the second column name (attribute) as defined in the workpiece class.

Throws:

java.lang.Exception
Administrative Actions

All of the methods described in this section are associated with various administrative type actions that are required by Enterprise Workflow. In general, these methods will allow you to create, change or delete configuration details such as workbaskets, users and workpiece classes.

Workbasket Maintenance

This section describes those methods associated the creation and maintenance of Enterprise Workflow workbaskets.

updateWorkbasket

Description

Use this method to change the attributes of a named workbasket.

Syntax

updateWorkbasket(java.lang.String name, java.lang.String accessList, java.lang.String forwardFlag, java.lang.String forwardToWorkbasket, java.lang.String notifyFlag, java.lang.String notifyMailAddress)

Parameters:

name - a string value that represents the name of the workbasket to update/change attributes.

accessList - a string value that represents the name of the access list to be associated with the workbasket.

forwardFlag - a string value of either 0 or -1 that represents if new items placed in the workbasket are to be forwarded to another workbasket. A 0 (zero) value enables the forward feature.

forwardToWorkbasket - a string value that represents the name of the workbasket to forward workpieces to when the forward flag is true.

notifyFlag - a string value of either 0 or -1 that represents if a notification message should be sent when workpieces arrive in the workbasket. A 0 (zero) value enables the notification feature.

notifyMailAddress - a string value that represents the email address(s) to use when sending the workpiece arrival message.

Returns:

0 if successful; -1 otherwise

Throws:

java.lang.Exception
deleteWorkbasket

Description
Use this method to remove a workbasket that was assigned to Enterprise Workflow.

Syntax
```java
deleteWorkbasket(java.lang.String name)
```

Parameters:
- name - a string value that represents the name of the workbasket to be deleted.

Returns:
- 0 if successful; -1 otherwise

Throws:
- java.lang.Exception

createWorkbasket

Description
Use this method to create a new "named" workbasket and associate an access list to it.

Syntax
```java
createWorkbasket(java.lang.String name, java.lang.String accessList, java.lang.String forwardFlag, java.lang.String forwardToWorkbasket, java.lang.String notifyFlag, java.lang.String notifyMailAddress)
```

Parameters:
- name - a string value that represents the new workbasket name.
- accessList - a string value that represents the access list name to associate with the workbasket.
- forwardFlag - a string value of either 0 or -1 that represents if new items placed in the workbasket are to be forwarded to another workbasket. A 0 (zero) value enables the forward features.
- forwardToWorkbasket - a string value that represents the name of the workbasket to forward workpieces to when the forward flag is true.
- notifyFlag - a string value of either 0 or -1 that represents if a notification message should be sent when workpieces arrive in the workbasket. A 0 (zero) value enables the notification feature.
- notifyMailAddress - a string value that represents the email address(es) to use when sending the workpiece arrival message.

Returns:
0 if successful; -1 otherwise

Throws:
   java.lang.Exception

Class Maintenance
This section describes those methods associated with the creation and maintenance of Enterprise Workflow workpiece class definitions.

updateClass
Description
   Use this method to update attributes of a workpiece class.

Syntax
   updateClass(java.lang.String name, java.lang.String accessList, java.lang.String description)

Parameters:
   name - a string value that represents the name of the class to update.
   accessList - a string value that represents the name of the access list to be associated with the class.
   description - a string value that is long text description of the class.

Returns:
   0 if successful; -1 otherwise

Throws:
   java.lang.Exception

deleteClass
Description
   Use this method to remove a workpiece class that had been defined in Enterprise Workflow.

Syntax
   deleteClass(java.lang.String name)

Parameters:
   name - a string value that represents the name of the workpiece class to be deleted.

Returns:
   0 if successful; -1 otherwise
Throws:

java.lang.Exception

**createClass**

**Description**

Use this method to define a new workpiece class within Enterprise Workflow.

**Syntax**

```java
createClass(java.lang.String name, java.lang.String accessList, java.lang.String description)
```

**Parameters:**

- **name** - a string value that represents the name of the new class.
- **accessList** - a string value that represents the name of the accessList to be associated with the class.
- **description** - a string value that represents the long text description of the class.

**Returns:**

0 if successful; -1 otherwise

Throws:

java.lang.Exception

**getWorkpieceClassColumnList**

**Description**

Use this method to retrieve a list of column names for a particular class. This list can be used to organize the name/value pairs into columns on the WorkbasketTOC so the user can readily identify workpieces by common attributes.

**Syntax**

```java
getWorkpieceClassColumnList(java.lang.String classID)
```

**Parameters:**

- **classID** - a string value that represents the name of the workpiece class to retrieve the associated column list from.

**Returns:**

An Object[] array of column name strings associated with the specified workpiece class.

Throws:

java.lang.Exception
**setWorkpieceClassColumnList**

**Description**
Use this method to set a list of column names for a particular class. This list can be used to organize the name/value pairs into columns on the WorkbasketTOC so the user can readily identify workpieces by column attributes.

**Syntax**

```java
setWorkpieceClassColumnList(String classID, Object[] list)
```

**Parameters:**
- `classID` - a string value that represents the name of the class.
- `list` - an Object[] array of column name strings.

**Returns:**
- 0 if successful; -1 otherwise

**Throws:**
- java.lang.Exception

**User Maintenance**
This section describes those methods associated with the creation and management of Enterprise Workflow users.

**createUser**

**Description**
Use this method to define a new user for Enterprise Workflow.

**Syntax**

```java
createUser(String name, String description, String accessList, String password)
```

**Parameters:**
- `name` - a string value that represents the user name identifier.
- `description` - a string value that represents the a more complete description of the user; for example, both first and last name of the new user.
- `accessList` - a string value that represents the access list.
- `password` - a string value that represents the user password for the profile.

**Returns:**
- 0 if successful; -1 otherwise

**Throws:**
- java.lang.Exception
removeUser

Description
Use this method to delete the definition of a particular workflow user.

Syntax
removeUser(java.lang.String name)

Parameters:
name - a string value that represents the user name to remove from the workflow environment.

Returns:
0 if successful; -1 otherwise

Throws:
java.lang.Exception

User Group Maintenance
This section describes those methods associated with the creation and maintenance of Enterprise Workflow user groups.

removeGroup

Description
This method removes the definition of a user group defined to the workflow environment.

Syntax
removeGroup(java.lang.String name)

Parameters:
name - a string value that represents the name of the user group to delete.

Returns:
0 if successful; -1 otherwise

Throws:
java.lang.Exception

createGroup

Description
Use this method to define a new user group for Enterprise Workflow.

Syntax

`createGroup(java.lang.String name, java.lang.String description)`

Parameters:

- **name** - a string value that represents the user group identifier.
- **description** - a string value that represents the long text description of the user group.

Returns:

- 0 if successful; -1 otherwise

Throws:

- java.lang.Exception

**getUserGroupMemberList**

Description

This method returns a list of all the members of a particular user group.

Syntax

`getUserGroupMemberList(java.lang.String name)`

Parameters:

- **name** - a string value that represents the name of the user group to retrieve the member list from.

Returns:

An Object[] array of user names and information.

- User Group Name - list of user group names.
- Member List - an array of user names associated with the user group.

```
<table>
<thead>
<tr>
<th>Element 1</th>
<th>{Element 2 - 0 to x elements}</th>
</tr>
</thead>
<tbody>
<tr>
<td>User Group Name</td>
<td>User 1</td>
</tr>
<tr>
<td></td>
<td>User 2</td>
</tr>
<tr>
<td></td>
<td>User x</td>
</tr>
</tbody>
</table>
```

Throws:
java.lang.Exception

**replaceUserGroupMemberList**

**Description**

Use this method to replace the members of a user group with a new list of users.

**Syntax**

```java
replaceUserGroupMemberList(java.lang.String name, java.lang.Object[] list)
```

**Parameters:**

- `name` - a string value that represents the name of the user group to replace the member list.
- `list` - an object array that represents the new list of users to be included as members of the specified user group.

**Returns:**

0 if successful; -1 otherwise

**Throws:**

java.lang.Exception

**Process Maintenance**

This section describes those methods associated with creation and maintenance of Enterprise Workflow processes.

**makeProcess**

**Description**

Use this method to create a workflow process from a deep array. See `getSteps()` for a description of the array.

**Syntax**

```java
makeProcess(java.lang.Object[] processSchema)
```

**Parameters:**

- `processSchema` - is the Object[] array containing a representation of the steps and connections that comprise the process. Refer to `getSteps()` for details of the array contents.

**Returns:**

0 if the process is successfully created.

**Throws:**

java.lang.Exception
**getSteps**

**Description**
Use this method to retrieve a deep array of all the steps and connections that comprise a workflow process.

**Syntax**

```java
getSteps(java.lang.String wfid)
```

**Parameters:**
No parameters associated with this method.

**Returns:**
A deep array Object[] that contains workflow process details; the array has four elements.

*Element 1 - Process ID* - represents the desired process ID to associate with the workpiece.

*Element 2 - Process Name* - a textual description of the process name.

*Element 3 - Process Description* - represents the long text description of the process name.

*Element 4 - Is another four-element array with each of the steps in the process.*

- **Step ID** - a unique value for each step in the process.
- **Current Step Type** - represents the type of step the workpiece is at. For example, a collection point or a workbasket.
- **Current Step Name** - a string value that represents the name of the step as defined in the workflow process.
- **Element 4 - Is another four-element array that contains the option details for the step.**

  - **Option ID** - a unique value that identifies a specific option for the step.
  - **Option Value** - a string that represents the text entered into the option parameter when defining a process.
  - **Option String** - a string that represents any parameters that are to be sent to an external program as defined in the option. Only necessary when associated with Exit Point steps.
  - **Next Step** - an identifier for the next step in the process.
### Lists

This section describes those methods associated with the retrieval of the various Enterprise Workflow lists; e.g., Access List list or a list of currently defined User Groups.

#### getAccessListList

**Description**

Use this method to retrieve a list of the names of all the access lists defined to the system.

**Syntax**

```java
getAccessListList()
```

**Parameters:**

No parameters associated with this method.

**Returns:**

Object[] array that contains the list of access lists.

**Throws:**

java.lang.Exception
java.lang.Exception

**getUserGroupList**

**Description**

Use this method to retrieve a list of user groups currently defined to the system.

**Syntax**

```java
getUserGroupList()
```

**Parameters:**

No parameters associated with this method.

**Returns:**

An Object[] array that represents the list of current user group names.

**Throws:**

java.lang.Exception

**getProcessList**

**Description**

This method returns an array naming all the workflow processes defined to the system.

**Syntax**

```java
getProcessList()
```

**Parameters:**

No parameters associated with this method.

**Returns:**

An Object[] array with three elements of process id, name, and description for each defined workflow process.

**Throws:**

java.lang.Exception

**getWorkbasketList**

**Description**

This method returns a list naming all the workbaskets defined to the workflow environment.

**Syntax**

```java
getWorkbasketList()
```
Parameters:

No parameters associated with this method.

Returns:

An Object[] array of workbasket names and descriptions.

- Workbasket - represents the workbasket name.
- Description - represents the long text description associated with the given workbasket.

Throws:

java.lang.Exception

**getClassList**

Description

Use this method to retrieve a list of all the workpiece classes defined within Enterprise Workflow.

Syntax

```java
getClassList()
```

Parameters:

No parameters associated with this method.

Returns:

Returns an Object[] array of class details; each array element contains an array of Strings describing a class.

Throws:

java.lang.Exception

**getUserList**

Description

This method returns a list of users defined to the workflow environment.

Syntax

```java
getUserList()
```

Parameters:

No parameters associated with this method.

Returns:

An Object[] array of currently defined users.
Security

All of the methods described in this section are associated with the various security functions of Enterprise Workflow.

User Authentication

This section describes those methods associated with validating Enterprise Workflow users.

authenticateUser

Description

This method tests the user and password passed through the HTTP headers. On failure it will throw a user not found exception error when the user is not good.

Syntax

\[
\text{authenticateUser()} \]

Parameters:

There are no parameters associated with this method.

Throws:

java.lang.Exception

User Authority

This section describes those methods associated with managing the given authorities (privileges) of Enterprise Workflow users.

userIsAuthorized

Description

Use this method to determine the users authorization to run a specified method. This method returns true if the user is permitted to use the method named in the privilege parameter. The user name is passed in the HTTP headers.

Syntax

\[
\text{userIsAuthorized(java.lang.String privilege)} \]

Parameters:

- \text{privilege} - a string value that represents the method to check.

Returns:
true if successful; false otherwise

Throws:
java.lang.Exception

userIsAuthorizedToWorkbasket

Description
Use this method to determine the user's authority to run a specified method that relates to a workbasket or relates to a workpiece currently in a workbasket. This method returns true if the user is permitted to use the method named in the privilege parameter. The user name is passed in the HTTP headers.

Syntax
userIsAuthorizedToWorkbasket(java.lang.String workbasket, java.lang.String privilege)

Parameters:
workbasket - a string value that represents the name of the workbasket to check privileges for.

privilege - a string value that represents the method to check.

Returns:
true if successful; false otherwise

Throws:
java.lang.Exception

userIsAuthorizedToClass

Description
Use this method to determine the user's authority to run a specified method against workpieces of the given class. This method returns true if the user is permitted to use the method named in the privilege parameter. The user name is passed in the HTTP headers.

Syntax
userIsAuthorizedToClass(java.lang.String className, java.lang.String privilege)

Parameters:

className - a string value that represents the name of the class.

privilege - a string value that represents the method to check.

Returns:
true if successful; false otherwise
Access List Maintenance

This section describes those methods associated the creation and maintenance of Access Lists.

getAccessListInfo

Description

This method returns the users and privileges associated through a particular access list.

Syntax

getAccessListInfo(java.lang.String accessListName)

Parameters:

accessListName - a string value that represents the name of the access list to retrieve details for.

Returns:

A deep Object[] of users and the associated privileges of each user.

- User - contains list of users currently associated with the access list.
- Privilege - an array of privileges assigned to the given user.

Element 1 |Element 2 - 1 to x privileges
---|---
User 1 | Privilege 1
| Privilege 2
| Privilege x

getAccessListInfo Array Structure

Throws:

java.lang.Exception

removeAccessList

Description

This method removes an access list that was defined to the workflow environment.

Syntax

removeAccessList(java.lang.String accessListName)
Parameters:

accessListName - a string value that represents the name of the access list to delete.

Returns:

0 if successful; -1 otherwise

Throws:

java.lang.Exception

createAccessList

Description

This method defines a new access list to the workflow environment. An access list associates a list of users with a list of privileges.

Syntax

createAccessList(java.lang.String accessListName, java.lang.String description,
java.lang.Object[] details)

Parameters:

accessListName - a string value that represents the name of the access list to be created.

description - a string value that represents a long description string for the access list.

details - a deep array Object[] that represents users to be associated with the access list; for each user there is an array of privileges associated with that user. The user may be an individual or a group.
<table>
<thead>
<tr>
<th>Access List Name</th>
<th>Description</th>
<th>User 1/User Group 1</th>
<th>Privilege 1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>Privilege 2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Privilege x</td>
</tr>
<tr>
<td>User 2/User Group 2</td>
<td></td>
<td></td>
<td>Privilege 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Privilege 2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Privilege x</td>
</tr>
<tr>
<td>User x/User Group x</td>
<td></td>
<td></td>
<td>Privilege 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Privilege 2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Privilege x</td>
</tr>
</tbody>
</table>

*createAccessList Array Structure*

**Returns:**

0 if successful; -1 otherwise

**Throws:**

java.lang.Exception

**getAccessListDetail**

**Description**

This method returns a list of the privileges assigned to the user through the access list.

**Syntax**

`getAccessListDetail(java.lang.String accessList, java.lang.String userName)`

**Parameters:**

accessList - a string value that provides the name of the access list you are interested in retrieving assigned privilege list.
**userName** - a string value that represents the user name to retrieve associated assigned privileges.

**Returns:**

Returns an Object[] array of privileges assigned to the user.

- User - contains list of users currently associated with the access list.
- Privilege - an array of privileges assigned to the given user.

<table>
<thead>
<tr>
<th>Element</th>
<th>(Element 2 - 1 to x privileges)</th>
</tr>
</thead>
<tbody>
<tr>
<td>User</td>
<td>Privilege 1</td>
</tr>
<tr>
<td></td>
<td>Privilege 2</td>
</tr>
<tr>
<td></td>
<td>Privilege x</td>
</tr>
</tbody>
</table>

**getAccessListDetail Array Structure**

**Throws:**

java.lang.Exception

**replaceAccessListDetail**

**Description**

This method replaces the privileges for a user in an access list with the new details provided.

**Syntax**

```
replaceAccessListDetail(java.lang.String accessList, java.lang.String userName, java.lang.Object[] details)
```

**Parameters:**

- accessList - a string value that represents the name of the access list to replace user privilege list.
- userName - a string value that represents the name of the user whose privilege list will be changed.
- details - an object[] array that lists the new list of privileges to be assigned to the specified user.

**Returns:**

0 if successful; -1 otherwise

**Throws:**

java.lang.Exception

**Privileges**

This section describes those methods associated with user privileges.
**getPrivilegeList**

**Description**

This method returns a list of the available webservice methods.

**Syntax**

```
getPrivilegeList()
```

**Parameters:**

No parameters associated with this method.

**Returns:**

An Object[] array of string values that represent each of the available webservice methods.

**Throws:**

java.lang.Exception
Create/Advance Workflow Workpieces 56
createAccessList 144
createClass 132
createGroup 134
createUser 133
createWorkbasket 130
CreateWorkpiece 118

- D -
Decision Point 48, 73, 75, 84
Decision Point Expressions 78
Default Path 78, 84
Default User ID 32
Define an Access List 36
Define Workpiece Class Details 41
deleteClass 131
deleteWorkbasket 130
Design a Workflow Process 48
DOCEXITC 109
DOCHKIN 109
Document Analysis 26

- E -
Evaluation Order 78
Exit 85
Exit Point 73, 75, 80, 85
Exit Point Class Definition 80
Exit Program 109, 118
Expressions 78

- F -
Find a Workpiece 63
Flow Chart 23
Forward 92

- G -
getAccessListDetail 145
getAccessListInfo 143
getAccessListList 138
getActiveWorkpieceList 128
getClassList 140
getPrivilegeList 147
getProcessList 139

getSteps 137
Getting Started Tasks 31
getUserGroupList 139
getUserGroupMemeberList 135
getUserList 140
getWorkbasketList 139
getWorkbasketTOC 126
getWorkpieceClassColumnList 132
getWorkpieceHistory 125
getWorkpieceInfo 119
getWorkpieceListByValues 127
getWorkpieceNotes 121

- H -
Help Desk 87

- I -
Implementation Plan 29
iSeries 10, 11

- J -
Java Runtime 10
JDBC 12
JSP 73
JVM 10

- L -
Login 76, 85
Login to Enterprise Workflow 32

- M -
Maintain Access List 36, 94
Maintain User Account 33, 97
Maintain User Group 68, 98
Maintain Workbaskets 46, 92
Maintain Workpiece Class 41, 93
makeProcess 136
Members 68, 98
Microsoft Database Engine 10
Move Workpiece 104
moveWorkpiece 124
Index

- N -
  Netserver 12
  New 83
  NFS 12
  Non-Members 68, 98
  Notes 76
  Notify 92

- O -
  OCR 26
  ODBC Data Source 12
  Option Edit Box 48
  Ordinary Workbasket User 36, 87

- P -
  Password 97
  Priority 56, 76
  Private User Workbasket 89
  Privileges Allowed for User 36, 94
  Process 75
  Process Analysis 23
  Process Designer 87
  Process Elements 73

- R -
  References 76
  removeAccessList 143
  removeGroup 134
  removeUser 134
  removeWorkpieceItem 123
  removeWorkpieceValue 123
  replaceAccessListDetail 146
  replaceUserGroupMemberList 136
  Retrieve 82
  Rollout Plan 29

- S -
  Sample Workflow 5
  Save 82
  Save As 82
  Scanning Documents 26
  Search 63, 76
  Search for Workpiece 63, 99
  Search for Workpieces 85
  setWorkpieceClassColumnList 133
  SQL Server 10, 12
  Start Point 73
  Stop Point 73, 74, 85
  Stop Workpiece 105
  stopWorkpiece 124
  Support 3
  System Requirements 10

- T -
  TCP/IP 11, 12
  Timeout Path 81
  Timeout Value 75
  Timeout Path 75
  Tomcat 10, 11, 12

- U -
  Unassigned Privileges 36, 94
  updateClass 131
  updateWorkbasket 129
  User Analysis 25
  User Group Name 98
  User Name 33, 97
  userIsAuthorized 141
  userIsAuthorizedToClass 142
  userIsAuthorizedToWorkbasket 142

- V -
  Variable Name 48, 84

- W -
  Web Service Menu 85
  WebDocs 12, 20, 28, 108, 109
  WebSphere 10, 11, 12
  Work Point 74, 76
  Work Point Options 77
  Workbasket 48, 73, 74, 84
Workbasket Name  46, 92
Workbasket User Interface  85
Workbasket User/Supervisor  87
Workflow Designer  48, 73, 77
Workflow Engine  73, 74, 75, 82
Workflow Process  77
Workflow Security  86
Working with Workbaskets  85
Workpiece  75, 76, 85
Workpiece Attributes  56, 91
Workpiece Class  41, 56, 91, 99
Workpiece Details  56
Workpiece History  103
Workpiece ID  103
Workpiece Reference  56
Workpiece References  91
Workpiece Security Class  75