# Sun Java™ System RFID Software 3.0 Administration Guide

Sun Microsystems, Inc.
www.sun.com

Part No. 819-4685-10
February 2006, Revision A

Submit comments about this document at: http://www.sun.com/hwdocs/feedback

Adobe PostScript™

# Contents

# Before You Begin

This administration guide for Sun Java™ System RFID Software 3.0 (RFID software) provides an overview of the RFID software architecture and instructions for configuring the various components. This guide is designed to aid system administrators and engineers who configure, administer and deploy the RFID software subsystems and components. This guide is intended primarily for Professional Services staff who are involved in promoting RFID applications to a broader audience through the EPCGlobal network.

Screen captures vary slightly from one platform to another. Although almost all procedures use the interface of the RFID software components, occasionally you might be instructed to type a command at the command line.

## Before You Read This Book

You should be familiar with RFID concepts and the following topics:

- Jini™ network technology concepts
- Java™ programming and concepts
- Java DataBase Connectivity (JDBC™) technology concepts and usage
- Java 2 Platform, Enterprise Edition (J2EE™ platform) technology and usage
- Client-server programming model
- Familiarity in managing large enterprise systems
- Administration of a supported application server
- Administration of a supported database

# Related Third-Party Web Site References

Third-party URLs are referenced in this document and provide additional, related information.

**Note –** Sun is not responsible for the availability of third-party web sites mentioned in this document. Sun does not endorse and is not responsible or liable for any content, advertising, products, or other materials that are available on or through such sites or resources. Sun will not be responsible or liable for any actual or alleged damage or loss caused or alleged to be caused by or in connection with use of or reliance on any such content goods or services that are available on or through such sites or resources.

# Documentation Formatting Conventions

This section describes the general conventions and typographic conventions used throughout this guide.

## General Conventions

The following general conventions are used in this guide:

- File and directory paths are given in UNIX® format (with forward slashes separating directory names).
- URLs are given in the following format:

  `http://`*server.domain*`/`*path*`/`*file*`.html` where TABLE P-1 describes the variables

**TABLE P-1** Definition of Variables Used in URLs

| Variable | Description |
|---|---|
| *server* | The name of the server (machine) where applications are run |

**TABLE P-1** Definition of Variables Used in URLs *(Continued)*

| Variable | Description |
|---|---|
| *domain* | Your Internet domain name |
| *path* | The server's directory structure indicating the path to the individual file |
| *file* | The individual file name |

- UNIX specific descriptions throughout this manual also apply to the Linux operating system, except where Linux is specifically mentioned.
- RFID installation root directories are indicated by the variable *rfid-install-dir* in this document. See Appendix C of the *Sun Java System RFID Software 3.0 Installation Guide* for more details.

# Typographic Conventions

| Typeface | Meaning | Examples |
|---|---|---|
| `AaBbCc123` | The names of commands, files, and directories; onscreen computer output | Edit your `.cvspass` file.<br>Use `DIR` to list all files.<br>`Search is complete.` |
| **`AaBbCc123`** | What you type, when contrasted with onscreen computer output | `>` **`login`**<br>`:` |
| *AaBbCc123* | Book titles, new words or terms, words to be emphasized | Read Chapter 6 in the *User's Guide*.<br>These are called *class* options.<br>You *must* save your changes. |
| *AaBbCc123* | Command-line variable; replace with a real name or value | To delete a file, type **`DEL`** *filename*. |

# Related Documentation

The following table lists the tasks and concepts that are described in the Sun Java System RFID Software manuals and release notes. If you are trying to accomplish a specific task or learn more about a specific concept, refer to the appropriate manual.

| Topic | For Information |
|---|---|
| Late-breaking information about the software and the documentation | *Sun Java System RFID Software 3.0 Release Notes* |
| Installing Sun Java System RFID Software and its various components. | *Sun Java System RFID Software 3.0 Installation Guide* |
| The following administration topics:<br>• RFID Software overview<br>• Configuring the RFID Event Manager<br>• Configuring Communication with SAP AII<br>• Using the RFID Management Console<br>• Configuring the RFID Information Server<br>• RFID device adapter reference<br>• RFID Event Manager component reference | *Sun Java System RFID Software 3.0 Administration Guide* |
| The following topics for RFID software developers:<br>• RFID Event Manager overview<br>• Creating custom filters and connectors<br>• Using RFID Device client APIs<br>• Using web services for device access<br>• Using Application Level Event (ALE) web services API<br>• Using RFID Information Server client APIs<br>• PML utilities | *Sun Java System RFID Software 3.0 Developer's Guide* |

# Sun Welcomes Your Comments

Sun is interested in improving its documentation and welcomes your comments and suggestions. Email your comments to Sun at this address: `docfeedback@sun.com`.

Include the part number (819-4685-10) of this document in the subject line of your email.

CHAPTER 1

# Sun Java System RFID Software Introduction

An overview of the Sun Java™ System RFID Software 3.0 is contained in the *Sun Java System RFID Software 3.0 Installation Guide*. You must be familiar with that introductory material before using this guide. The RFID Software is supported on several operating system (OS) platforms, application servers and databases. Refer to the *Sun Java System RFID Software 3.0 Release Notes* for the latest details.

This chapter includes:

- RFID Electronic Product Code Details
- RFID Information Server

## RFID Electronic Product Code Details

The RFID Software uses the identification scheme for Electronic Product Code (EPC) data formatting as defined in the specification *EPC_Tag Data Standards Version 1.1 Rev.1.27*.

This section includes the following topics:

- EPC Data Format
- EPC Network

### EPC Data Format

An EPC functions similarly to a Universal Product Code (UPC) as found in common bar code technology. The EPC is an identification scheme for universally identifying physical objects using Radio Frequency Identification (RFID) tags and other means. The standardized EPC data format consists of an EPC (sometimes called an EPC

Identifier) that uniquely identifies an individual object, and may include an optional Filter Value when it is necessary to enable the effective and efficient reading of the EPC tags.

The EPC encoded in an RFID tag can identify the manufacturer, product, version, and serial number. The EPC also provides an extra set of digits to identify unique items.

The major part of the standard EPC data field is the EPC Identifier. The optional Filter Value field within the EPC can supplement the basic EPC tag readings. For various applications and industries, the EPC Version 1.1 standard specifies the following coding schemes:

- General Identifier (GID)
- A serialized version of the EAN.UCC Global Trade Item Number (GTIN)
- EAN.UCC Serial Shipping Container Code (SSCC)
- EAN.UCC Global Location Number (GLN)
- EAN.UCC Global Returnable Asset Identifier (GRAI)
- EAN.UCC Global Individual Asset Identifier (GIAI)

For any given RFID tag with an EPC data format, an entry in its header field indicates which coding scheme can be applied.

## Layered Concepts

Independent from underlying physical media, such as RFID tags or bar codes, a *pure identity* represents a unique entity in an abstract form. The EPC standard provides this definition for a pure identity: "The identity associated with a specific physical or logical entity, independent of any particular encoding vehicle such as an RF tag, bar code, or database field."

The EPC standard further defines *identity URI* as "a representation of a pure identity as a Uniform Resource Identifier (URI). A URI is a character string representation that is commonly used to exchange identity data between software components of a larger system."

The standard URI representation of EPCs has four categories:

- URIs for pure identities (also called *canonical forms*), which contain only the EPC fields to identify a physical object. For example, a pure identity URI for GID can be "urn:epc:id:gid:10.1002.2". A URI for GRAI can be "urn:epc:id:grai:0652642.12345.1234."
- URIs for EPC tags, which represent the tag encodings. These URIs can be used by application software to write a tag. An example for a serialized GTIN 64-bit encoding is "urn:epc:tag:sgtin-64:3.0652642. 800031.400."
- URIs for raw bit strings, which represent invalid bit-level patterns as a single decimal number. For example: "urn:epc:raw:64.20018283527919."

- URIs for EPC patterns, which refer to a set of EPCs for the purpose of EPC filtering. A pattern of "urn:epc:pat:sgtin-64:3.0652642.[1024-2047].*" refers to any SGTIN Identifier 64-bit tag with a filter value of three, a company prefix of 0652642, an item reference in the range from 1024 to 2047 and any serial number.

An *encoding* identity layer can also be conceptualized and would comprise a pure identity together with additional information such as the filter value, rendered into a specific syntax (typically consisting of value fields of specific sizes). A given pure identity may have a number of possible encodings, such as a barcode encoding, various tag encodings, and various URI encodings. Encodings may also incorporate additional data besides the identity such as the filter value used in some encodings, in which case the encoding scheme specifies what additional data it can hold.

Finally, a Physical Realization of an Encoding, is an encoding rendered in a concrete implementation suitable for a particular machine-readable form (such as a specific kind of radio frequency tag or specific database field). This encoding can be conceived as a lower layer, like the ISO's Open System Interconnect with its modeling of physical entities near the bottom of the stack.

## EPC Network

The RFID Software consists of the RFID Event Manager and the RFID Information Server modules. The RFID Event Manager gathers information from RFID readers, filters the information, and provides the processed information to the RFID Information Server module or to a third-party Enterprise Resource Planning (ERP) system.

The following illustration shows how the RFID Event Manager and RFID Information Server fit into the EPC network.

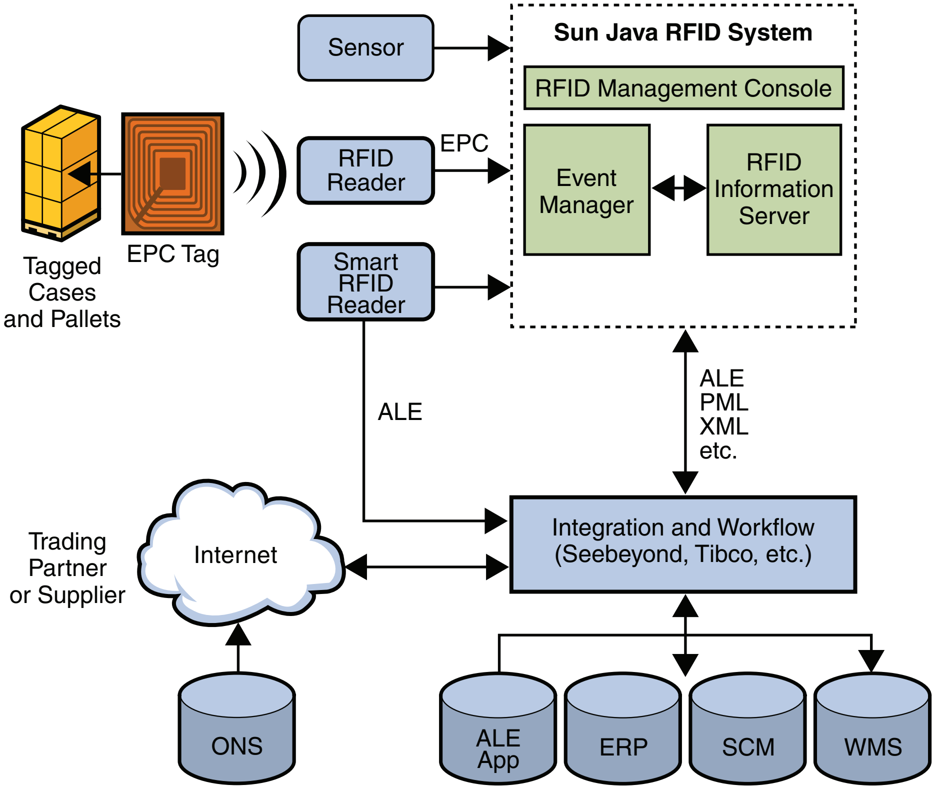


**FIGURE 1-1** Sun Java System RFID Software in the EPC Network

# RFID Information Server

The RFID Information Server is a Java 2 Platform, Enterprise Edition (J2EE™) application that serves as an interface for capture and query of EPC-related data. EPC-related data can include tag observation data from RFID Event Manager as well as information that maps EPCs to higher-level business data. The RFID Information Server is typically used to translate a set of low-level observations into higher-level business functions.

Other applications interface with the RFID Information Server through XML message exchange. The RFID Information Server supports HTTP and Java™ Message Service (JMS) technology message transports. All data persists in a relational database. You can find the supported software listed in the *Sun Java System RFID Software 3.0 Release Notes*.

The RFID Software uses the JMS API as one of its primary methods for communicating with third-party software. Communication with the RFID Information Server is stateless and synchronous. If HTTP is used as the transport, the client uses `HTTP POST` to communicate with the RFID Information Server. To implement synchronous requests with JMS API, the client uses a message ID to correlate requests with responses. JMS messages, requests and responses, are posted to a well-known topic. See "Enabling Usage of JMS With the RFID Information Server" on page 106.

For developer convenience, a Java client library to programatically access the RFID Information Server is provided. The API to query and manipulate data in the RFID Information Server is independent of the protocol used. See the *Sun Java System RFID Software 3.0 Developer's Guide* for API information.

CHAPTER 2

# Configuring the RFID Event Manager

This chapter describes the necessary concepts and procedures for configuring the RFID Event Manager using the RFID Configuration Manager tool.

This chapter includes the following topics:

- Event Manager Concepts
- RFID Configuration Manager Overview
- Using the RFID Configuration Manager
- Managing Device Profiles
- RFID Event Manager Component Overview

## Event Manager Concepts

The RFID Event Manager consists of a Control Station and one or more Execution Agents as described in the *Sun Java System RFID Software 3.0 Installation Guide*. Each Execution Agent is composed of an adapter that passes information into one or more filters or connectors. The filters, in turn, pass information into one or more connectors. This chain of processes constitutes a federation of services, known as *Business Processing Semantics* (BPS), where each of the services contains one or more components linked together to process events. Using the RFID Configuration Manager, you implement a BPS by creating a configuration object.

An RFID reader is a piece of hardware that communicates with RFID tags using a microwave radio frequency. The RFID Reader communicates with a device-specific adapter through a proprietary protocol. The device adapter is a driver-like piece of Java code that interfaces to the actual RFID reader device. The supported device adapters are listed in TABLE 2-3. Device profiles for the supported adapters are provided with the RFID Software.

The adapter functions by receiving the EPC of the RFID tag and generating an event that includes a timestamp and the source of the event. The source of the event is the reader and antenna that read the tag. The event is posted to a set of listeners, such as filters or connectors, that then process the event.

Filters can *smooth* the data by throwing away previously detected information, or by using a mask-matching algorithm to route information to other components based on the mask criteria. The filtered data is posted as an event to defined listeners. The listeners can be other filters that further process the data or connectors that serve as connectors to third-party applications that use the RFID tag event information. This collection of adapters, filters and connectors is known as a role. The following diagram shows a sample component arrangement using the adapter for the Mercury3 reader that is passing data into two separate roles.

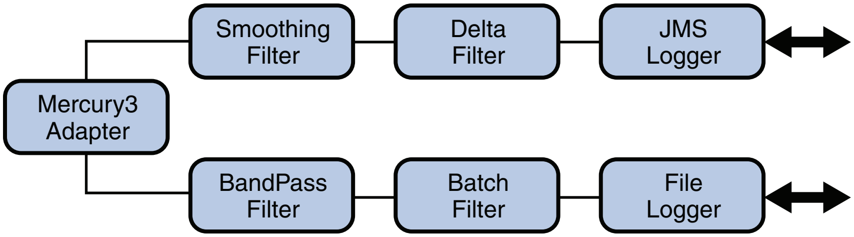


**FIGURE 2-1** Sample BPS/Role Component Arrangement

# RFID Configuration Manager Overview

In the first release of this software, the configuration of the Business Processing Semantics (BPS) for an Execution Agent was accomplished by editing the XML configuration file for each Execution Agent. Release 2.0 of the software provided a configuration tool, the RFID Configuration Manager, to simplify this task. You should not edit the raw XML files to define the adapter properties of the RFID readers or to define the BPS. You should use the RFID Configuration Manager tool to define a configuration object. The configuration object instantiates a specific BPS.

**Note –** See the appendixes for information on configuring the Event Manager by editing the XML configuration file. Appendix A contains information for the supported device adapters. Appendix B contains information for the supported filters and connectors. Appendix C contains sample XML files.

The RFID Configuration Manager consists of a configuration service and a user interface to the service. To begin using the RFID Configuration Manager, it is necessary to understand the concepts of device profiles, RFID Event Manager components, roles, configuration objects, and devices.

- **Device profiles –** A device profile provides the fully qualified description of a given RFID reader or other device (such as a printer or sensor) model and its properties. The device profile provides a sensor input point to the RFID system. Each model from a hardware vendor has a unique profile that describes it. For example, the Alien ALR-9780 profile defines that its reader has a maximum of four antennas, includes an `antennaSequence` property to define the order in which each antenna is probed, and includes a `username` and a `password` property for communication with the device, and so on. Default values are provided for each device profile, but it is in the device that the values are actually used.
- **Event Manager components** – A component is the basic building block for the RFID system. All components have inputs and outputs. You can connect the output of one component to the input of another component to create a processing chain. Two types of components are currently used by Sun's RFID Software:
  - **Filters –** A filter takes inputs and transforms them, according to an algorithmic process description, into an output.
  - **Connectors –** A connector provides a routing point from the process flow to an external or internal point. Examples of connectors are a Java Message Service (JMS) connector and a and file connector. A JMS connector places data onto a JMS queue. A file connector stores data into a file.

  See "RFID Event Manager Component Overview" on page 53 for more details on the components provided with the RFID Software.
- **Event Manager roles –** A role is a combination of components providing a functional capability within the RFID architecture. By themselves, roles are not functional, as they lack the ability to read data from the RFID system. For example, a role might represent a dock door, an assembly line station, or a smart shelf. A role consists of a chain of filters and connectors with a designated data input point. The data input point defines the RFID Event Manager component that initially receives and processes the tag event data from the RFID devices. The role specification replaces a large segment of the contents of the `RfidConfig.xml` configuration file.
- **Devices –** A device is a concrete physical instance of a particular profile. Multiple devices can be associated with a specific profile. These devices may or may not be identical.
- **Configuration objects –** A configuration object defines an actual distributed, active RFID data sensor that provides a RFID reader function. The configuration object is generally a role that receives input from multiple devices, each of which belongs to a specific profile. A configuration object associates a set of one or more physical devices and its respective device profiles to a role. For example, a

configuration object might associate the Alien reader with IP address 129.146.28.22 and a specific user name and password to the Alien-ALR-9780 device profile using the DockDoor role.

- For example, consider the case where you create two roles, 1=DockDoor and 2= ConveyorBelt. You can use these roles to create multiple configuration objects, depending on the location and functionality of the associated physical reader, such as the following:
  - DockDoor1a
  - DockDoor2a
  - ConveyorBeltInbound
  - ConveyorBeltOutbound

  One configuration object might indicate the inbound direction and another configuration object might indicate the outbound direction through the dock door depending on the property settings of the physical readers associated with the configuration object.

The RFID Configuration Manager is the tool used to create and manage the components, profiles, roles, devices and configuration objects. The next section contains procedures that describe how to use this tool.

# Using the RFID Configuration Manager

The RFID Configuration Manager interface is composed of the following elements:

- **Menu** – Contains all functions that are available in the RFID Event Manager.
- **Toolbar** – Contains frequent use functions that are depicted by icons.
- **Navigation tree** – Shows all components, device profiles, and roles at a glance.
- **Drawing pane** – Shows role structure and configuration objects.
- **Drawing pane icons** – see the following table for an explanation of each icon's function.

**TABLE 2-1** RFID Configuration Manager Drawing Icons

| Icon | Description |
|---|---|
| | Add a filter to a role. |
| | Add a connector to a role. |
| | Define an input point for a role. |

**TABLE 2-1** RFID Configuration Manager Drawing Icons *(Continued)*

| Icon | Description |
|---|---|
| ⊞ | Autolayout function redraws the diagram in a connection ordered fashion. Note: This icon has nothing to do with grouping or ungrouping functions. |
| ⧉ | Grouping function that enables you select multiple boxes and move them around as a unit. |
| ⧉ | Ungrouping function enables you move multiple boxes grouped as a unit one at a time again. |
| 🔍 | Zoom |
| 🔍+ | Zoom in |
| 🔍− | Zoom out |

Use the RFID Configuration Manager to perform the following configuration tasks:

- "To Define the RFID System Physical Devices" on page 25
- "To Define RFID Event Manager Roles" on page 30
- "To Define the RFID System Configuration Object" on page 38
- "To Edit a Device Profile" on page 51

A predefined set of device profiles and RFID Event Manager components are included with the software. These device profiles and components are necessary building blocks for configuring the RFID Software system. You can modify the existing components by adding properties or changing the value of existing properties. You can also specify new device profiles and components.

The basic steps for configuring your RFID Event Manager are shown in the following table.

**TABLE 2-2** Overview of RFID Event Manager Configuration Process

| Task | Procedure |
|---|---|
| 1. Install RFID Software Event Manager. The RFID Configuration Manager is installed as part of this installation. | *Sun Java System RFID Software 3.0 Installation Guide* |
| 2. Start the RFID Configuration Manager. | "Starting the RFID Configuration Manager" on page 22 |
| 3. Define the RFID system's physical readers. | "To Define the RFID System Physical Devices" on page 25 |

**TABLE 2-2** Overview of RFID Event Manager Configuration Process *(Continued)*

| Task | Procedure |
|---|---|
| 4. Use the Configuration Manager to create roles. | "To Define RFID Event Manager Roles" on page 30 |
| 5. Create the Event Manager Configuration Objects. | "To Define the RFID System Configuration Object" on page 38 |
| 6. Start the RFID Event Manager. | *Sun Java System RFID Software 3.0 Installation Guide* |

## Starting the RFID Configuration Manager

**Prerequisite** – If you have not installed the RFID Event Manager, see the *Sun Java System RFID Software 3.0 Installation Guide*.

You need to use the RFID Configuration Manager on the machine where the RFID Event Manager Control Station is installed. The RFID Configuration Manager needs access to the file system on that machine.

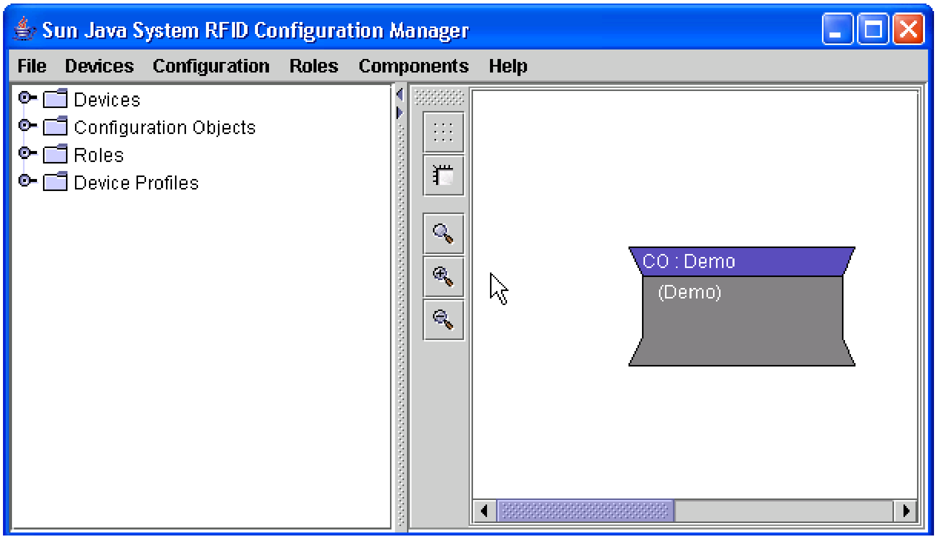
## ▼ To Start the RFID Configuration Manager (UNIX)

**Prerequisite** – You need superuser or administrator privileges to start the RFID Configuration Manager.

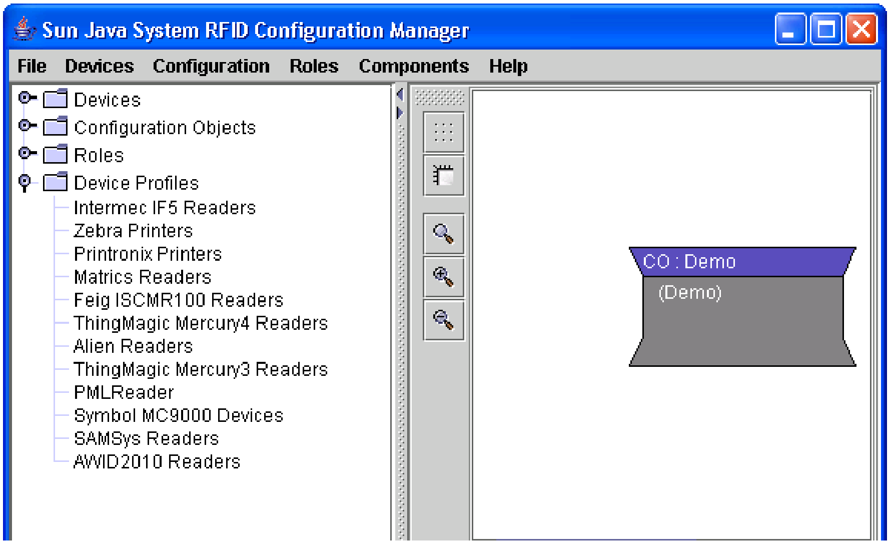
1. **Start the RFID Configuration Manager by using the** `rfidconfig` **script.**

   - **Solaris:** `/opt/SUNWrfid/bin/rfidconfig`
   - **Linux:** `/opt/sun/rfidem/bin/rfidconfig`

   The RFID Configuration Manager appears similar to the following screen capture. The nodes in the navigation tree on the left might be expanded.



2. **(Optional) Expand the Device Profiles node to see the supported device profiles that have been installed with your RFID Event Manager.**



Intermec IF5 Readers
Zebra Printers
Printronix Printers
Matrics Readers
Feig ISCMR100 Readers
ThingMagic Mercury4 Readers
Alien Readers
ThingMagic Mercury3 Readers
PMLReader
Symbol MC9000 Devices
SAMSys Readers
AWID2010 Readers

## ▼ To Start the RFID Configuration Manager (Microsoft Windows)

- Start Menu option – Choose Start → Programs → Sun Microsystems → Sun Java System RFID Software → Configuration Manager.
- Command window – `C:\Program Files\Sun\RFID Software\rfidem\bin\rfidconfig.bat`

## RFID Software Deployment Parameters

During installation, you specified a unique Jini™ group to use in your RFID environment. The RFID Configuration Manager Deployment Parameters dialog box enables you to change the value of this Jini group. You also use this dialog box to define the port number. By default, the web server serving the Java classes to the Execution Agents is Project Rio's Webster. The Execution Agents receive their workload dynamically at startup and obtain the Java classes required to execute the workload from this web server. This web server is configured at installation time to use port 52493. This dialog box displays the current system values.

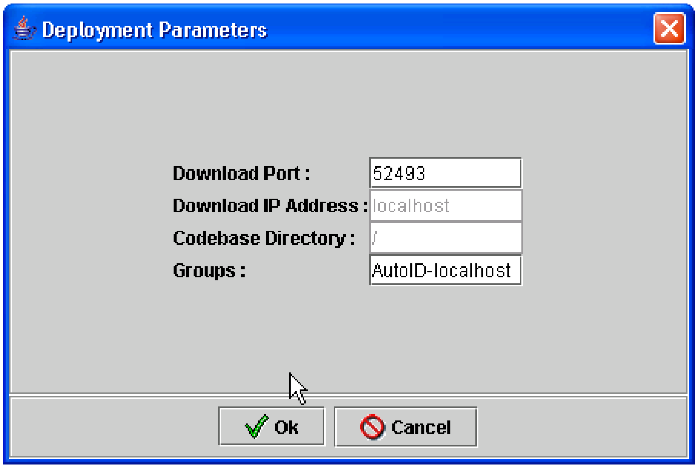
## ▼ To Review or Change the RFID Deployment Parameters

1. **Start the RFID Configuration Manager.**

   See "To Start the RFID Configuration Manager (UNIX)" on page 22.

2. **Choose File → Settings.**

   The Deployment Parameters dialog box appears.



Download Port : 52493
Download IP Address : localhost
Codebase Directory : /
Groups : AutoID-localhost

3. **Review or change the parameters as necessary.**

   You are discouraged from changing these parameters except in the following situations:

   - A conflict exists with the default port.
   - The initial Jini group is not unique.

---

**Caution –** Exercise extreme caution when updating these parameters, as the system can become unusable if the incorrect parameters are used.

---

## ▼ To Define the RFID System Physical Devices

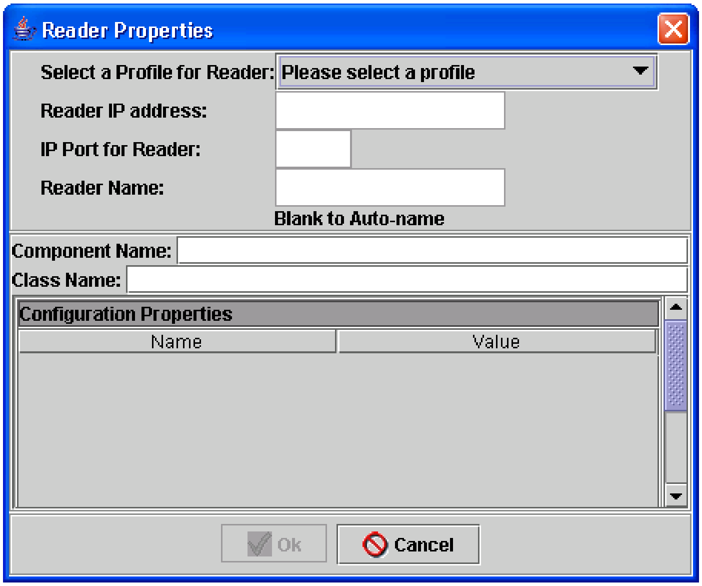
A *device instance* is a physical instance of a device profile in the RFID system. A physical device must have a defining device profile associated with it. All device instances are defined by their device profiles, IP addresses and IP ports. The device profile and IP address must be defined. The IP port is optional.

1. **If you have not already done so, start the RFID Configuration Manager.**

   See "To Start the RFID Configuration Manager (UNIX)" on page 22 or "To Start the RFID Configuration Manager (Microsoft Windows)" on page 24.

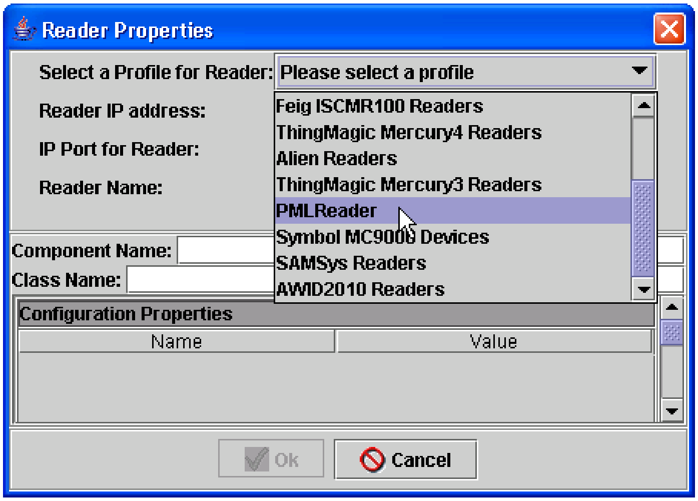
2. **From the RFID Configuration Manager menu, choose Devices → New.**

   The Reader Properties dialog appears.



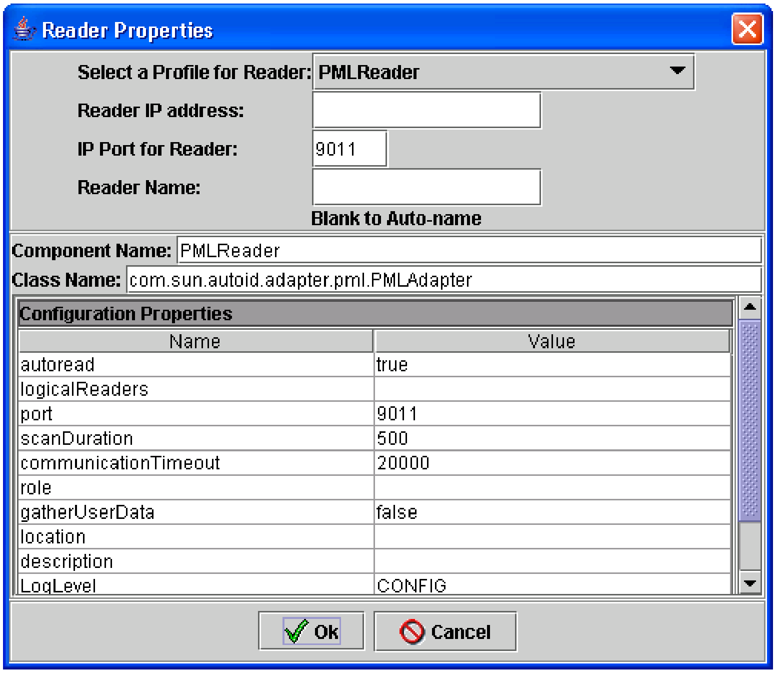
3. **Click Please Select a Profile to see a drop-down list of possible device profiles.**

   All other fields are null until you select the base profile.



4. **For the purposes of this example, select PMLReader from the drop-down list.**

   The default properties associated with the PMLReader are displayed in the dialog box.



Select a Profile for Reader: PMLReader

Reader IP address:

IP Port for Reader: 9011

Reader Name:

Blank to Auto-name

Component Name: PMLReader

Class Name: com.sun.autoid.adapter.pml.PMLAdapter

Configuration Properties

| Name | Value |
|---|---|
| autoread | true |
| logicalReaders | |
| port | 9011 |
| scanDuration | 500 |
| communicationTimeout | 20000 |
| role | |
| gatherUserData | false |
| location | |
| description | |
| LogLevel | CONFIG |

5. **Type the reader IP address, the IP port number, and the reader name.**

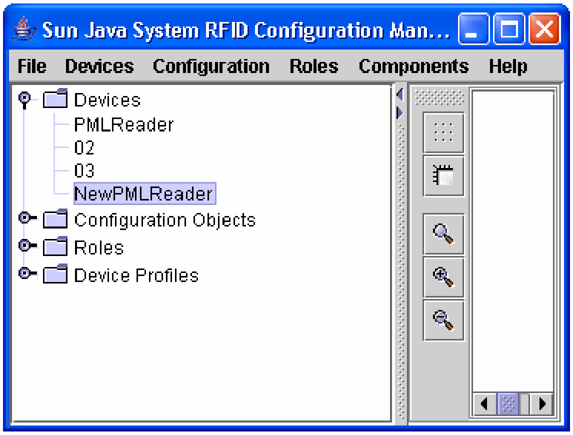
   Use the reader IP address and IP port number for an actual physical reader in your RFID system. These property values associate the physical reader to the appropriate device profile. Use a meaningful name, as this reader name appears in the navigation tree under the Devices node.

   For the purposes of this example, type the following:

   - Reader IP address – `129.135.15.2`
   - IP Port for Reader – 2005
   - Reader Name – NewPMLReader

6. **(Optional) Change the configuration property values as needed. When finished, click Ok.**

   The newly created device instance appears in the navigation tree as shown in the following screen capture.



7. **(Optional) Add any new necessary configuration properties.**

   See "To Add a Configuration Property to a Reader" on page 28.

## ▼ To Add a Configuration Property to a Reader

**Prerequisite** – This procedure assumes that you have created the example `NewPMLReader` as described in the previous procedure, "To Define the RFID System Physical Devices" on page 25.

1. **If you have not already done so, start the RFID Configuration Manager.**

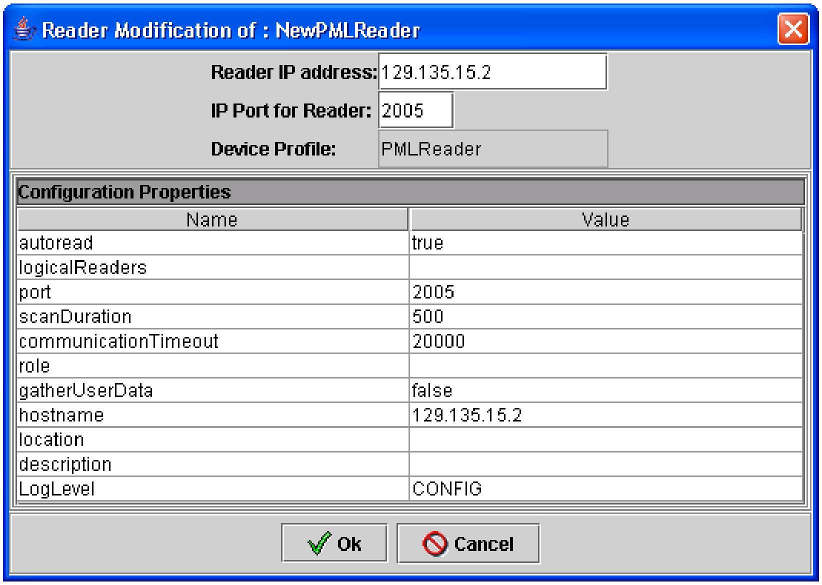
   See "To Start the RFID Configuration Manager (UNIX)" on page 22 or "To Start the RFID Configuration Manager (Microsoft Windows)" on page 24.

2. **From the RFID Configuration Manager menu, choose Devices → Edit.**

   A dialog listing the available devices appears.

3. **For the purposes of this example, select NewPMLReader (created in the previous procedure) and click Ok.**

   An edit dialog box for the device appears as shown in the following screen capture.


Reader Modification of : NewPMLReader

Reader IP address: 129.135.15.2

IP Port for Reader: 2005

Device Profile: PMLReader

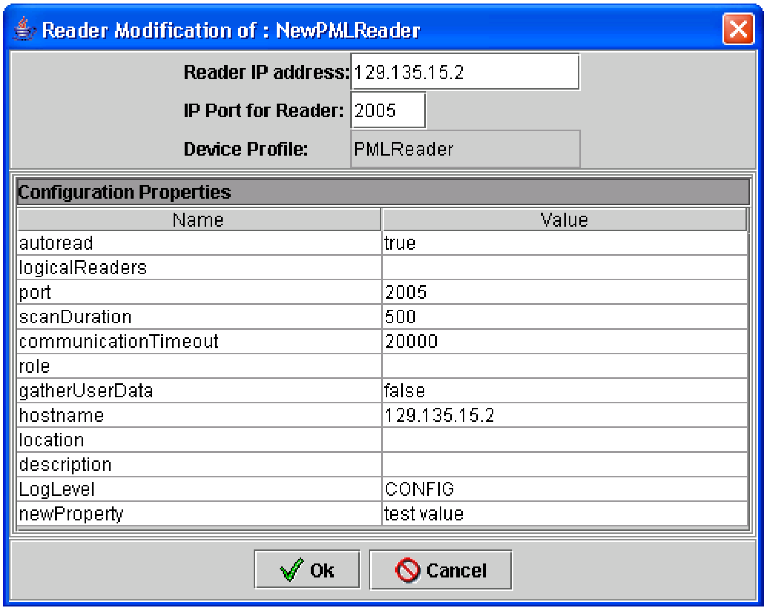
Configuration Properties

| Name | Value |
|---|---|
| autoread | true |
| logicalReaders | |
| port | 2005 |
| scanDuration | 500 |
| communicationTimeout | 20000 |
| role | |
| gatherUserData | false |
| hostname | 129.135.15.2 |
| location | |
| description | |
| LogLevel | CONFIG |

4. **Select any configuration property name field.**
5. **Right-click and choose Add Property from the contextual menu.**

   A blank row appears enabling you to add the Name and Value for the new property.
6. **Type the appropriate values and click Ok.**

   The new configuration property is added to the device. The following screen capture shows shows the newly added configuration property of `newProperty` added to the NewPMLReader.



Reader Modification of : NewPMLReader

Reader IP address: 129.135.15.2

IP Port for Reader: 2005

Device Profile: PMLReader

Configuration Properties

| Name | Value |
|---|---|
| autoread | true |
| logicalReaders | |
| port | 2005 |
| scanDuration | 500 |
| communicationTimeout | 20000 |
| role | |
| gatherUserData | false |
| hostname | 129.135.15.2 |
| location | |
| description | |
| LogLevel | CONFIG |
| newProperty | test value |

## ▼ To Define RFID Event Manager Roles

Use the drawing pane of the RFID Configuration Manager to visually create the roles that comprise your RFID application.

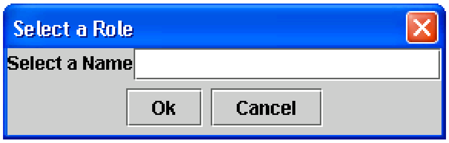
The role created in the following procedure detects RFID tags from a reader, *smooths* the tag distribution, and records the detected RFID tags in a file on every cycle. Depending on the physical environment, RFID tags can appear and disappear from the reader's view at rapid intervals. This role might be used in such an environment because it smooths the true appearance of tags over time. For more information on the smoothing function, see "Smoothing Filter" on page 168.

1. **If you have not already done so, start the RFID Configuration Manager.**

   See "To Start the RFID Configuration Manager (UNIX)" on page 22 or "To Start the RFID Configuration Manager (Microsoft Windows)" on page 24.

2. **Choose Roles → New.**

   A small dialog box appears.



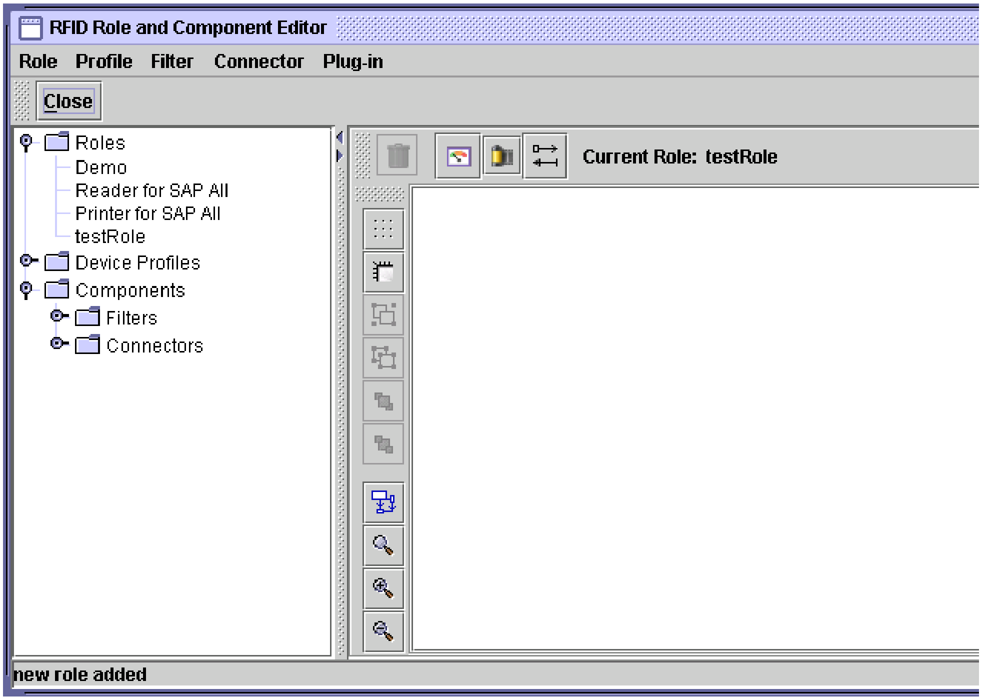
Select a Role

Select a Name

3. **In the Select a Name field, type the name of the Role and click Ok.**

   Use a descriptive name for the role. For this example, use the name `testRole` as shown in the following screen capture.

   A new window opens and displays the RFID Role and Component Editor and its drawing pane. The drawing pane is prepared to create the new role in the Role and Component Editor. The new role name appears in the navigation tree on the left of the designer under the Roles node.

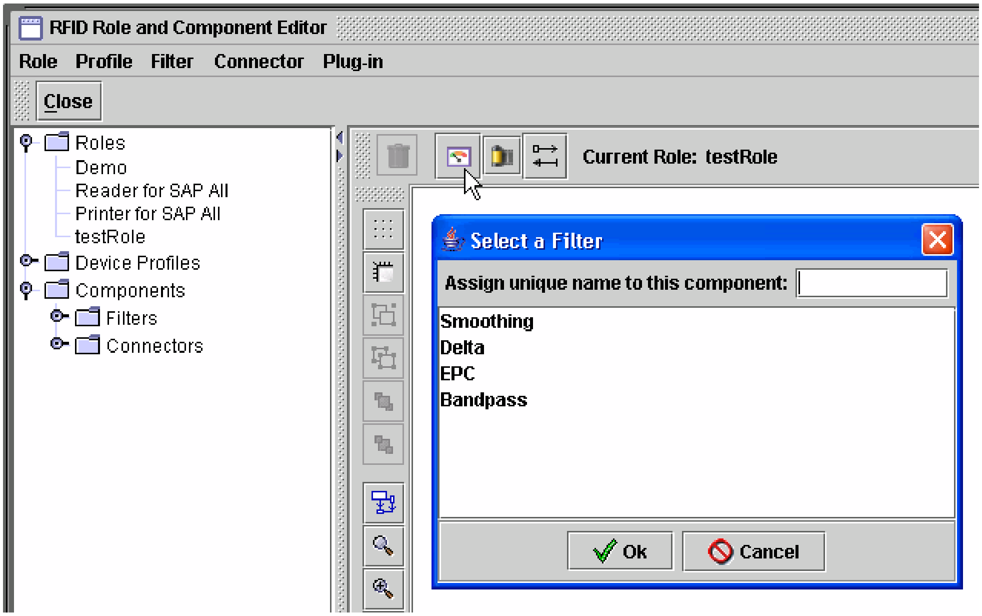


Current Role: testRole

new role added

4. **Define the role by using the icons at the top of the drawing pane.**

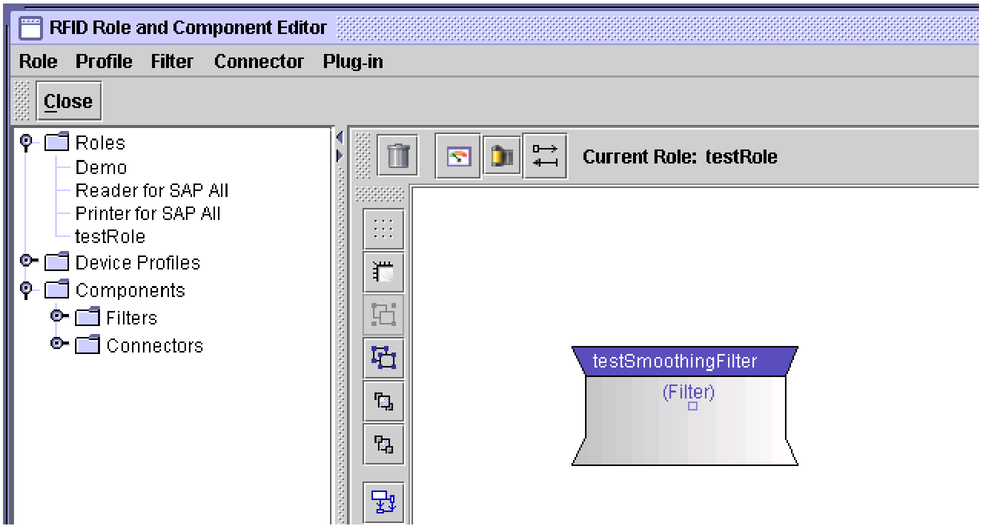
   Click the filter icon (looks like a meter panel), to see a drop-down list similar to the following screen capture. Using the icons, you add filters and connectors to the role.



5. **Select a component. For this example, select the Smoothing Filter.**

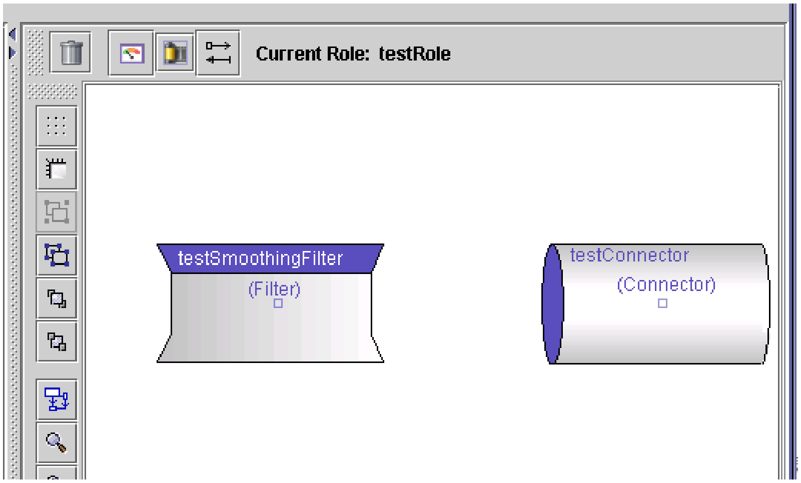
6. **Type a unique name for this filter, for example,** `testSmoothingFilter`**, and click Ok.**

   The filter appears on the drawing pane as shown in the following screen capture.



7. **Click the connector icon (looks like a roll of film) and add a File connector named** `testConnector`**.**

   The following screen capture shows the drawing pane after a smoothing filter and a file connector have been added.

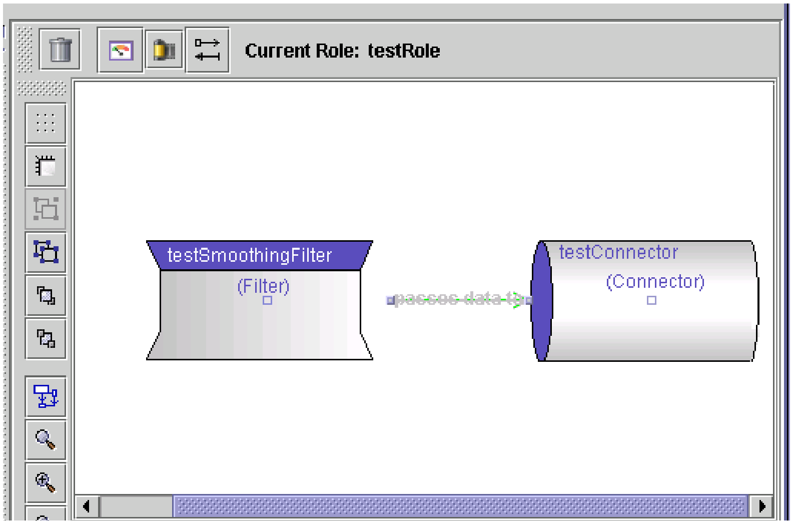


8. **Connect the components in the necessary order.**

   To do so, click the port (the small square at the center of each component) and drag a line to another component. This action connects the output of one component to the input of another component. Filters connect to connectors and to other filters.

**Caution –** Do not connect a connector to another connector, this configuration has no meaning in the RFID system configuration.

The following screen capture shows the `testSmoothingFilter` that is connected to the `testConnector`.



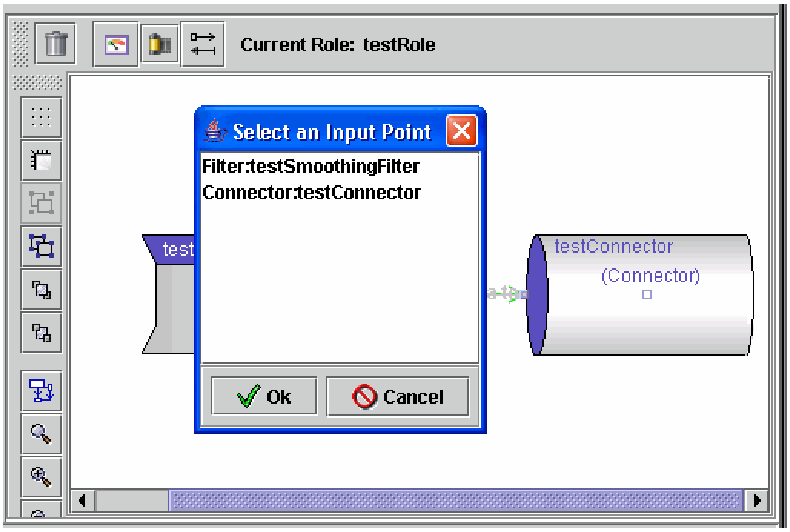
9. **Add an input point to the role.**

   The *input point* for the role is the attachment point by which a physical device provides data to the role in a real configuration instance. You must define an input point for the role.

   a. **Click the input icon.**

      This is the icon that has two arrows that point in opposite directions. Place the cursor over an icon to view a description of its function.

      A dialog box that shows the available input points appears. These input points are the components that you have added to this role. Typically, the input point is a filter.



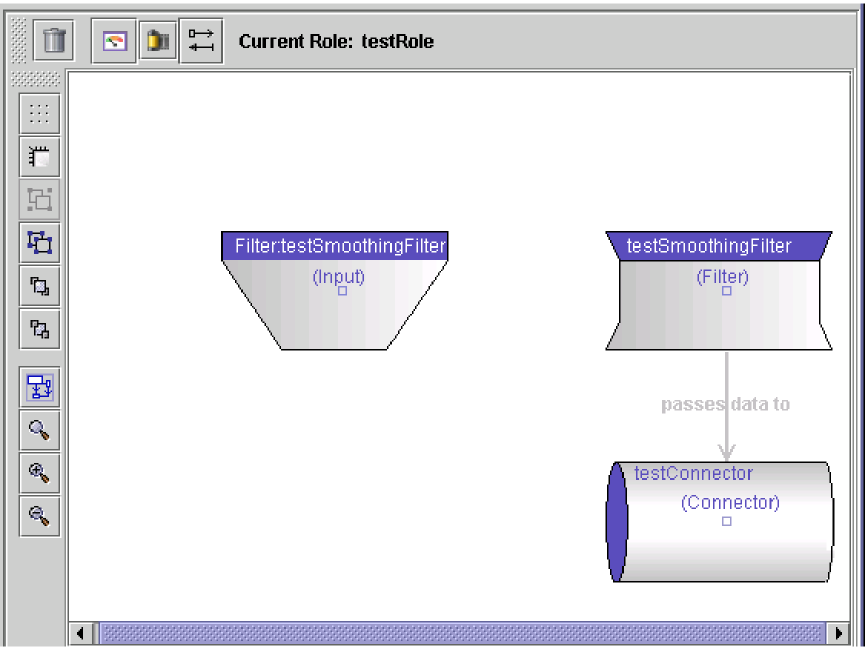
Current Role: testRole

Select an Input Point

Filter:testSmoothingFilter
Connector:testConnector

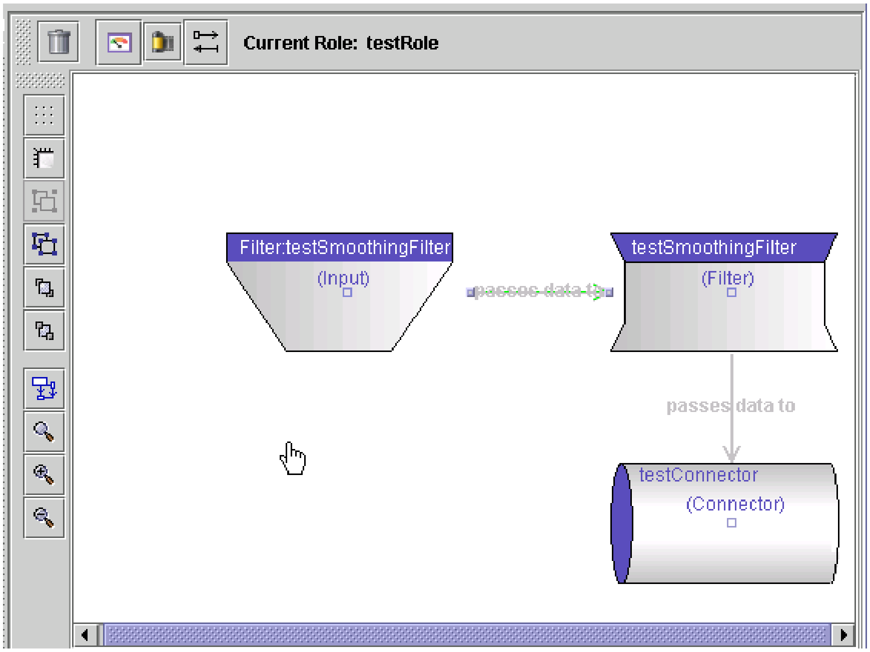
b. **Select the input point and click Ok to add it to the role.**

The input point appears on the drawing pane.



c. **Connect the input point to the appropriate component.**

Once you have added the necessary filters, connectors, an input point and connected everything as needed, the role is ready to save. The following screen capture shows a completed role with one filter, one connector, and the filter designated as the input point that directs the input data to the connector.



10. **To save the role, return to the RFID Configuration Manager.**

    When you started this procedure to define a role, a second window opened. The title bar of this second window is labeled RFID Role and Component Editor. Return the focus of your cursor to the original window with the title bar labeled RFID Configuration Manager.

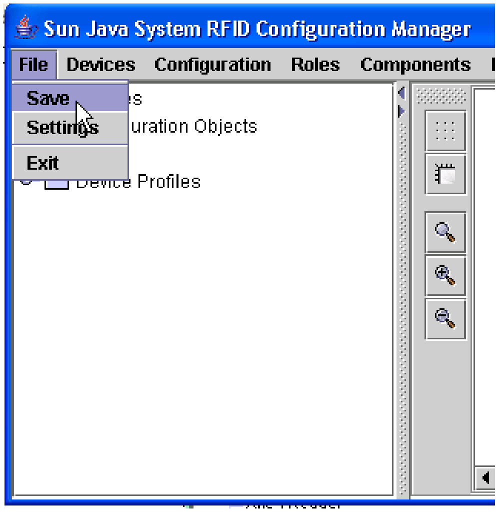
---

**Caution –** You cannot save the role from the RFID Role and Component Editor window. Do not close the RFID Configuration Manager window without first saving your work.

---

11. **From the RFID Configuration Manager's main menu, choose File → Save.**

    This step is necessary to save all work done with the RFID Configuration Manager.



**Note –** Changes to a role after a configuration object is created do not propagate to the configuration objects.

For example, if you create a configuration object, `CODemo`, that instantiates the Demo role and then later, you add a new filter or connector to the Demo role, the `CODemo` does *not* include this new component. To use the updated Demo role, a new configuration object needs to be created. You can continue to use the original `CODemo` configuration object or it can be deleted.

## ▼ To Define the RFID System Configuration Object

Configuration objects are the active entities within the RFID system. They represent the union of the physical hardware and the software components that interact within an RFID implementation. When you have defined your physical readers as devices and have created the roles for your RFID system, you are ready to complete the configuration process by defining the *configuration objects*.

Each time you change a Configuration Object, the RFID Event Manager must be restarted in order for the changes to take effect. Wait until the system is completely started or you may lose tag events. Depending on the components you used in your role, the RFID Tag Viewer may or may not appear (it depends on the objects in your role). The delay lets the system come up completely before generating EPC tags.

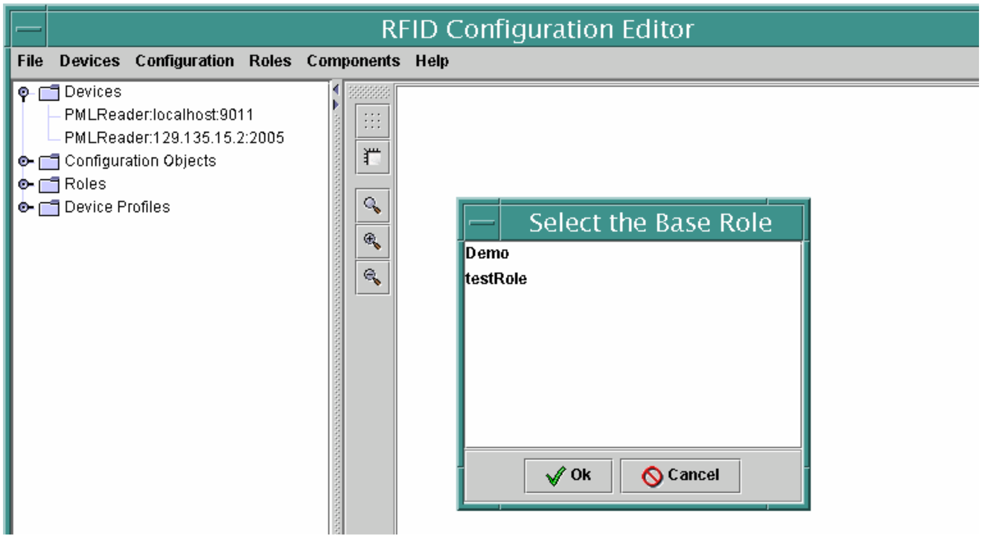
**Note –** Configuration objects cannot be reused for roles that have been modified. If you modify a role, you must define a new configuration object to use that role.

1. **If you have not already done so, start the RFID Configuration Manager.**

   See "To Start the RFID Configuration Manager (UNIX)" on page 22 or "To Start the RFID Configuration Manager (Microsoft Windows)" on page 24.

2. **From the RFID Configuration Manager menu, choose Configuration → New.**

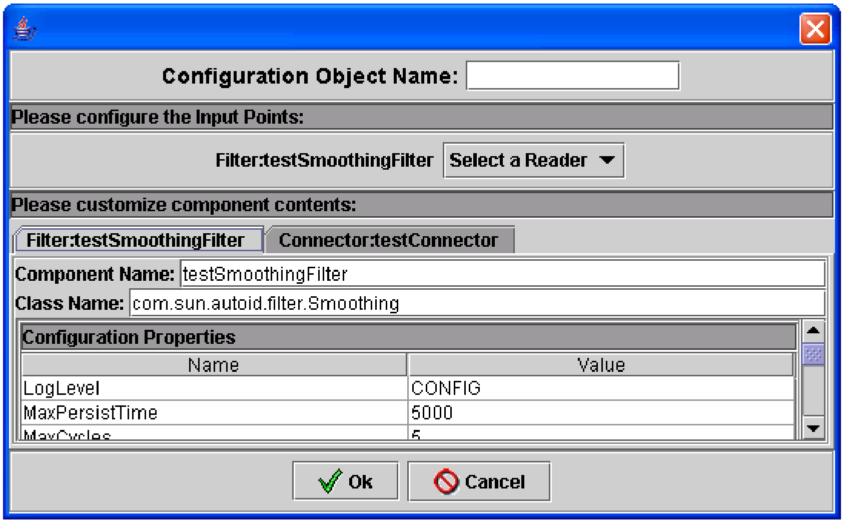
   A dialog box appears that lists the available roles.



3. **Select the role on which to base this configuration object, and click Ok.**

   The Configuration Object dialog box appears.

   For this example, select `testRole`, that was created in the previous procedure "To Define RFID Event Manager Roles" on page 30.



Configuration Object Name:

Please configure the Input Points:

Filter:testSmoothingFilter

Please customize component contents:

Component Name: testSmoothingFilter

Class Name: com.sun.autoid.filter.Smoothing

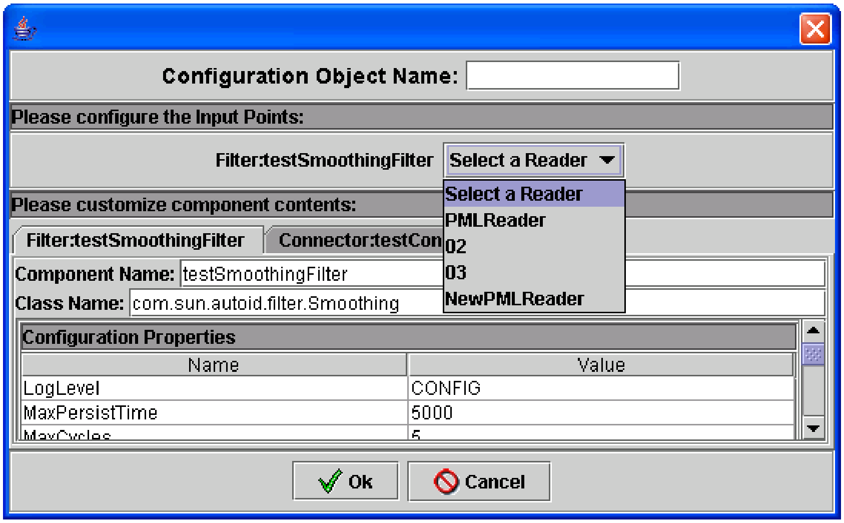
Configuration Properties

| Name | Value |
|---|---|
| LogLevel | CONFIG |
| MaxPersistTime | 5000 |
| MaxCycles | 5 |

This dialog box has four areas for input as follows:

- The Configuration Object Name field.
- The Input Point Configuration area, which contains a drop-down list of the available devices. Click Select a Reader to see the drop-down list.
- A tabbed section for customizing the role components that shows a separate tab for each component that is part of the role.
- The Configuration Properties area that corresponds to the selected component tab.

You can change component properties of the role to satisfy unique constraints of your system. For illustration purposes, the following screen capture shows the `testRole` and devices that were defined in the prior procedures with the list of defined devices expanded.



Configuration Object Name:
Please configure the Input Points:
Filter:testSmoothingFilter
Select a Reader
Select a Reader
PMLReader
02
03
NewPMLReader
Please customize component contents:
Filter:testSmoothingFilter
Connector:testCon
Ok
Cancel

Component Name: testSmoothingFilter

Class Name: com.sun.autoid.filter.Smoothing

Configuration Properties

| Name | Value |
|---|---|
| LogLevel | CONFIG |
| MaxPersistTime | 5000 |
| MaxCycles | 5 |

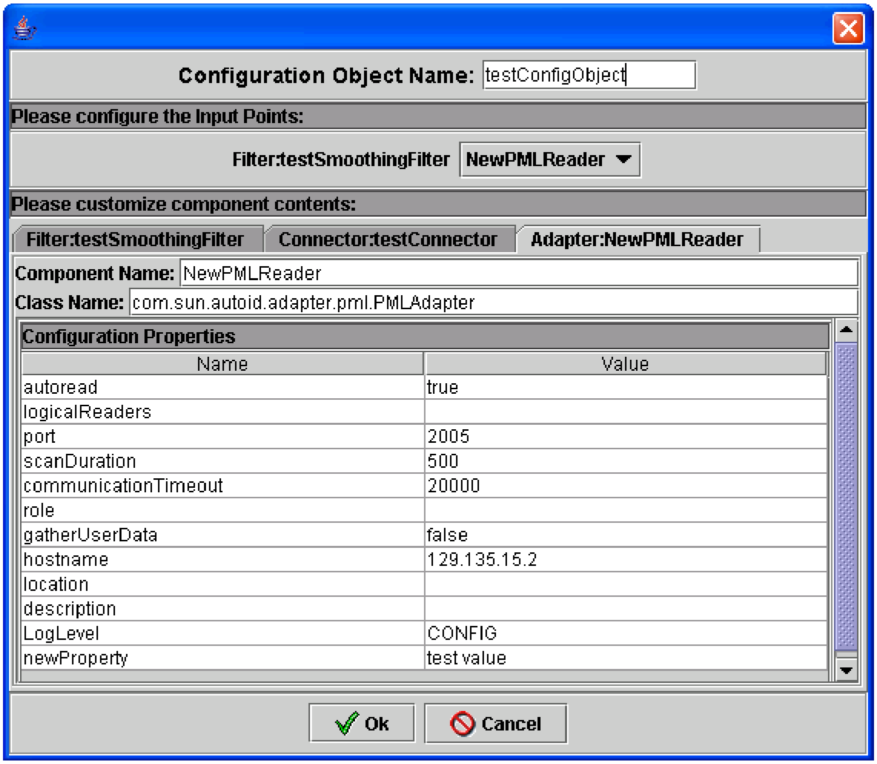
4. **In the Configuration Object Name field, type a name for your configuration object.**

5. **Select a reader to add that reader's component property set to the configuration object and propagate the appropriate interconnections.**

   In this example, select `NewPMLReader`. A tab is added showing the configuration properties for the selected reader.

6. **Click the `NewPMLReader` tab to show the reader properties.**

   You can make changes to the specific values for this configuration object on this tab. For example, you can change the maximum number of cycles that defines the `communicationTimeout` property for the reader.



Configuration Object Name: testConfigObject

Please configure the Input Points:

Filter:testSmoothingFilter NewPMLReader

Please customize component contents:

Component Name: NewPMLReader

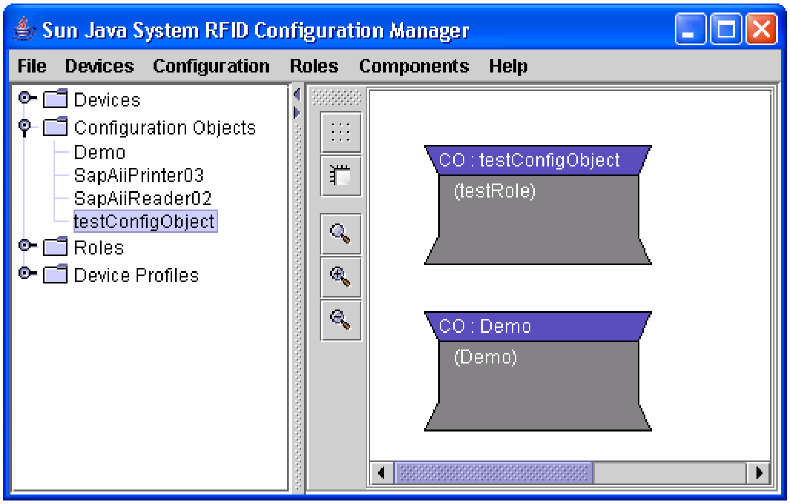
Class Name: com.sun.autoid.adapter.pml.PMLAdapter

Configuration Properties

| Name | Value |
|---|---|
| autoread | true |
| logicalReaders | |
| port | 2005 |
| scanDuration | 500 |
| communicationTimeout | 20000 |
| role | |
| gatherUserData | false |
| hostname | 129.135.15.2 |
| location | |
| description | |
| LogLevel | CONFIG |
| newProperty | test value |

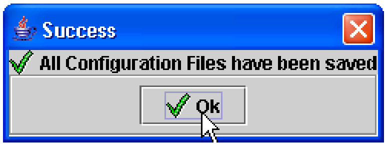
7. **When you have finished customizing the component properties, click Ok to create the configuration object.**

   The configuration object appears on the drawing pane of the RFID Configuration Manager and also appears in the navigation tree under the Configuration Objects node. The following screen capture shows the `testConfigObject` created in the prior steps.



8. **To save your work, choose File → Save from the RFID Configuration Manager main menu.**

   A confirmation message appears.



All Configuration Files have been saved

9. **Click Ok.**

   Your work is saved.

## ▼ To Modify a Configuration Object

1. **If you have not already done so, start the RFID Configuration Manager.**

   See "To Start the RFID Configuration Manager (UNIX)" on page 22 or "To Start the RFID Configuration Manager (Microsoft Windows)" on page 24.

2. **From the RFID Configuration Manager menu, choose Configuration → Edit.**

3. **Select the configuration object to edit.**

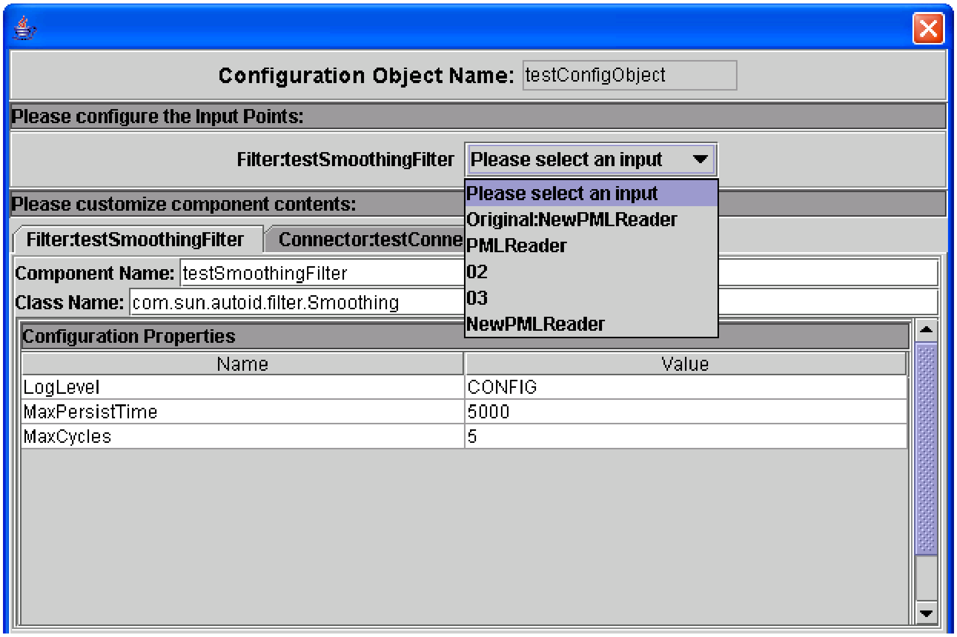
   You are prompted to indicate whether you want to change the devices that are assigned to this configuration object.

4. **Decide what to edit.**

   The dialog box that appears depends on whether you choose to edit the devices associated with this configuration object or the properties of the existing devices and components.

   - **To specify a new and different set of devices** (readers or printers) for the role, choose Yes and follow these steps.

   a. **Click Please Select an Input to see a drop-down list of devices.**



**Configuration Object Name:** testConfigObject

**Please configure the Input Points:**

Filter:testSmoothingFilter: Please select an input

**Please customize component contents:**

Component Name: testSmoothingFilter

Class Name: com.sun.autoid.filter.Smoothing

Configuration Properties

| Name | Value |
|---|---|
| LogLevel | CONFIG |
| MaxPersistTime | 5000 |
| MaxCycles | 5 |

   b. **Select the new device and click Ok.**

   c. **To save your work, choose File → Save from the RFID Configuration Manager main menu.**

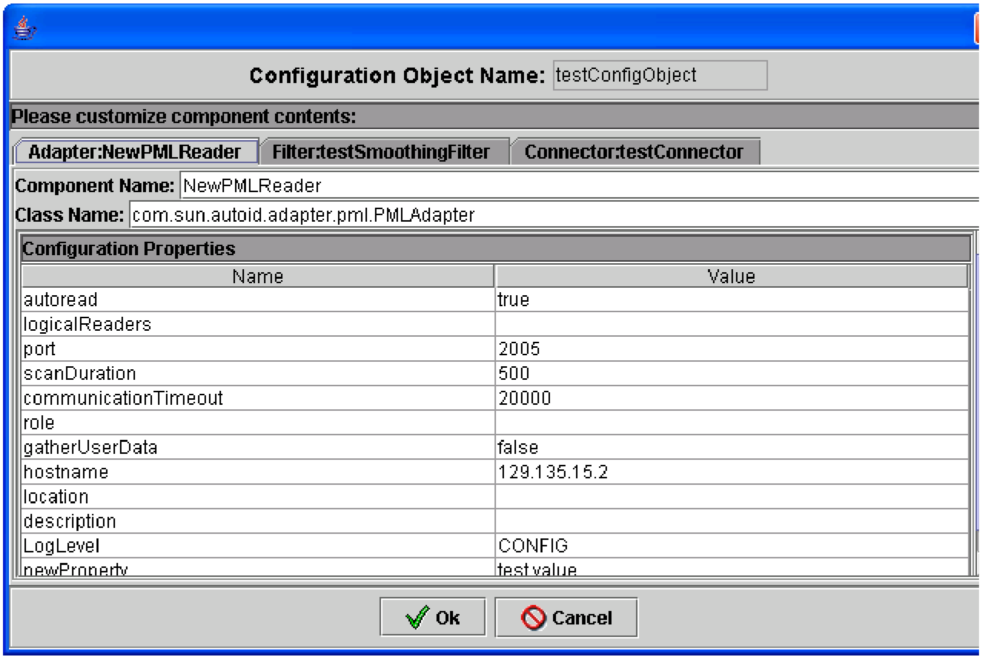
   A confirmation message appears.

   d. **Click Ok.**

   Your work is saved.

   - **To change the properties** of the previously assigned devices and components, choose No and follow these steps.

   a. **Select the tab for the component properties that you need to change.**



Configuration Object Name: testConfigObject

Please customize component contents:

Component Name: NewPMLReader

Class Name: com.sun.autoid.adapter.pml.PMLAdapter

Configuration Properties

| Name | Value |
|---|---|
| autoread | true |
| logicalReaders | |
| port | 2005 |
| scanDuration | 500 |
| communicationTimeout | 20000 |
| role | |
| gatherUserData | false |
| hostname | 129.135.15.2 |
| location | |
| description | |
| LogLevel | CONFIG |
| newProperty | test value |

**b. Change the properties as needed and click Ok.**

**c. To save your work, choose File → Save from the RFID Configuration Manager main menu.**

A confirmation message appears.

**d. Click Ok.**

Your work is saved.

# ▼ To Start the RFID Event Manager

After you have defined your RFID system's configuration objects, you must stop and restart the RFID Event Manager if it is running. If it is not running, start it. The following command line examples use the default *rfid-install-dir* for each platform.

- Solaris OS – `/opt/SUNWrfid/startall`
- Linux OS – `/opt/sun/rfidem/startall`

For the Microsoft Windows platform, use the Start menu option.

- Start Menu option, use Choose Start → Programs → Sun Microsystems → Sun Java System RFID Software → Start Agent [and Start Station].

If you use the Start menu option, you must select both Start Agent and Start Station.

- Command line option – You can also start both the Control Station and the Execution Agent by using the `startall` script found at `C:\Program Files\Sun\RFID Software\rfidem\bin.`

**Note –** Refer to the *Sun Java System RFID Software 3.0 Installation Guide* for more information on installed scripts and directories associated with this software.

# Managing Device Profiles

The RFID Software comes with a predefined set of device profiles that describe the supported device adapters. The adapters interpret the specific communication protocol used by the leading EPC-compliant RFID readers and devices such as printers. These device adapters can be viewed as equivalent to the device drivers that are used to communicate with peripherals in a computer system. Each adapter implements vendor-specific communication protocols to communicate with that vendor's RFID device.

The device profiles describe the set of properties and their values for the device adapters. When you start the RFID Configuration Manager, you have access to these device profiles. You can modify the default settings and use the device profiles to create the configuration objects that are needed by your RFID system.

The RFID Configuration Manager provides the following functions for managing RFID device profiles:

- New – create a new device profile
- Edit – customize the default device profiles
- Remove – delete a device profile

Device profiles implement adapters with a particular set of property values. Adapters gather RFID events from readers and propagate the events to other components as defined by the role. For this software release, the RFID Event Manager supports devices through the use of the adapters described in TABLE 2-3.

The following properties define an adapter:

- **name** – The unique name for the component.
- **classname** – The name of the Java class that implements the component.
- **properties** – A sequence of name-value pairs that are used for the configuration of the adapter. Properties common to all readers are listed in TABLE 2-4.

Device-specific properties and additional considerations for the common properties are described in the device-specific tables that are listed in the Adapter Configuration Description column of TABLE 2-3.

- **outputs** – A sequence of component names to be registered as event listeners to this adapter. The outputs normally designate one or more filters or connectors.

## Device Profiles for Supported RFID Devices

Device profiles for the readers and other devices listed in the following table are preloaded in the RFID Configuration Manager. You can modify the default settings by using the RFID Configuration Manager that is described in "Using the RFID Configuration Manager" on page 20.

**TABLE 2-3** Supported RFID Devices

| Reader or Other Device Manufacturer and Model | For Information about Adapter Configuration |
|---|---|
| AWID MPR-2010 Reader | See TABLE A-1. |
| Feig LRU1000 Reader | See TABLE A-2. |
| Feig Electronic ID ISC MR.100 Reader<br>Feig Electronic ID ISC PR.100 Reader | See TABLE A-3. |
| Intermec Intellitag IF5 Fixed Reader | See TABLE A-4. |
| Matrics RDR-001 Reader<br>Matrics AR-400 Reader | See TABLE A-9. |
| ThingMagic Mercury3 RFID Reader<br>Sensormatic SensorID Agile 1 Reader | See TABLE A-7. |
| ThingMagic Mercury4 RFID Reader<br>Sensormatic SensorID Agile 2 Reader | See TABLE A-8. |
| Alien ALR-9780 Reader<br>Alien NanoScanner 915 Mhz Reader | See TABLE A-6. |
| Software PML reader (intended to be used as a testing tool) | See TABLE A-5. |
| Printronix SL 5000e Printer | See TABLE A-10. |
| SAMSys MP9320 EPC V2.7 | See TABLE A-11. |
| Symbol MC9000-W Handheld Reader or any PocketPC with a network connection | See TABLE A-13. |
| Zebra Technologies R110XiIIIPlus Printer | See TABLE A-14. |

All device adapters support a basic set of properties, unless otherwise specified in the configuration section for the specific adapter. The common properties are described in the following table.

**TABLE 2-4** Device Adapter Common Properties

| Property | Description | Values |
|---|---|---|
| `LogLevel` | Specifies the detail of logging information to be generated by the adapter. This property overrides the system-wide logging level for the component.<br>Also see "Additional Information for Common Properties" on page 50. | In descending order the `LogLevel` values are:<br>• SEVERE<br>• WARNING<br>• INFO<br>• CONFIG<br>• FINE<br>• FINER<br>• FINEST<br>• OFF - turns off logging<br>• ALL - enables logging of all messages |
| `hostname` | Specifies the IP address or *hostname* of the network- based RFID reader. In the case of RFID readers connected to a serial Ethernet converter, *hostname* specifies the address of the Ethernet converter. | For example:<br>`192.168.1.25` |
| `port` | Specifies the port number where the RFID reader listens for connections from the RFID Event Manager. | For example: 9011 |
| `readerepc` | An EPC value associated with the reader that identifies it in the EPC network. This property should not be confused with the EPC values that the reader obtains from the tags.<br>The RFID Event Manager uses this value to identify the source of the read events.<br>Also see "Additional Information for Common Properties" on page 50. | For example:<br>`urn:epc:tag:gid-96:1.255.1` |

**TABLE 2-4** Device Adapter Common Properties *(Continued)*

| Property | Description | Values |
|---|---|---|
| `autoread` | Available on some readers and sed in conjunction with the property, `scanDuration`.<br>• When enabled, the reader is in a continuously *listening* mode in the following way – the reader gathers tag information for `scanDuration` msecs, reports the findings and loops again through the same read and report sequence.<br>• If `autoread` is set to false, then the filters and connectors that receive the events from the reader adapter see no events, unless external software initiates the tag list request programmatically.<br>Also see "Additional Information for Common Properties" on page 50. | Default value = false (disabled)<br>Value = true (enabled) |
| `communicationTimeout` | Indicates how long to wait for a reader response before retransmitting a read request. | Default value = 10,000 msecs<br>Using a higher value, such as 20 seconds, causes the adapter to generate fewer retransmissions, but potentially detects network problems less rapidly. |
| `scanDuration` | Indicates the antenna pulse duration in milliseconds. The antenna pulse duration is the time that the antenna is powered and actively scanning for RFID tags. | Default value = 500 msecs |
| For enabling antennae and discriminating between antennae varied properties exist that are specific to each reader adapter. | Each adapter has a unique way of enabling the various antennae, as well as discriminating between the multiple antennae of one reader. | N/A |

## Additional Information for Common Properties

- **Log level** – Log level settings follow logging conventions that are established in J2SE, version 1.4.2. for more information, see the API documentation for the class `java.util.logging.Level` found at `http://java.sun.com/j2se/1.4.2/docs/api/index.html`. Also see "Log Files" in the *Sun Java System RFID Software Installation Guide*.
- **Reader EPC identifier –** This property, `readerepc`, is the EPC value that is associated with the reader. This property is used by the RFID Event Manager to identify the source of the events. Values other than the value shown in TABLE 2-4 are possible. Refer to the EPC specification for further customization information.
- **Autoread property** – Some RFID readers can be configured to continuously report RFID tag events in their field of view, without requiring the adapter to constantly ask for them. When the `autoread` property is enabled, the adapter configures the RFID reader into automatic read mode and goes into a listening mode. In turn, the adapter reports the list of tags to its consumers (the filters and components that receive the tag event data). The PMLReader supports `autoread` mode.

  When the `autoread` property is enabled, the adapter instructs the reader to do the following:

  - Pulse its antennae for the `scanDuration` number of milliseconds.
  - Post the RFID tag events to the event consumers.
  - Loop again to gather new RFID tag events.

  When the RFID reader does not support automatic mode and the `autoread` property has been enabled, the adapter loops generating requests to the reader to obtain the list of tags. This setting generates extra network traffic, as there needs to be a request for every response, instead of simply listening for reports without asking for them.

  When the `autoread` property is disabled, the consumers of the reader adapter do not see tag events, unless an external software module initiates the tag list request programmatically. The `autoread` property is disabled by default.

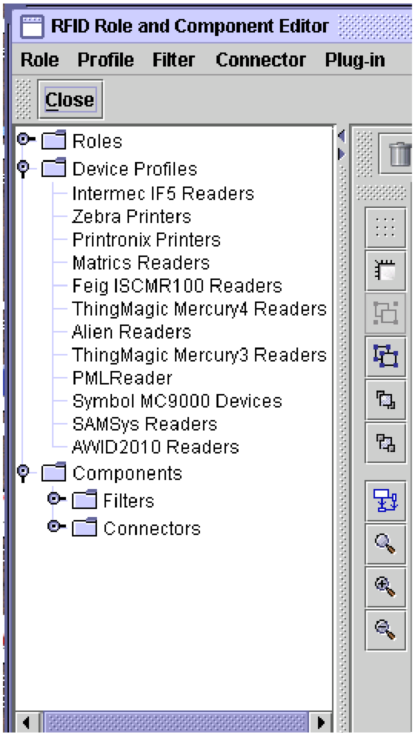
## ▼ To Edit a Device Profile

1. **If you have not already done so, start the RFID Configuration Manager.**

   See "To Start the RFID Configuration Manager (UNIX)" on page 22 or "To Start the RFID Configuration Manager (Microsoft Windows)" on page 24.

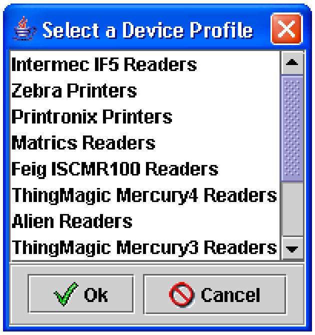
2. **From the RFID Configuration Manager main menu, choose Components → Device Profiles.**

   The RFID Role and Component Editor appears with the Device Profiles node expanded.



3. **From the RFID Role and Component Editor main menu, choose Profile → Edit.**

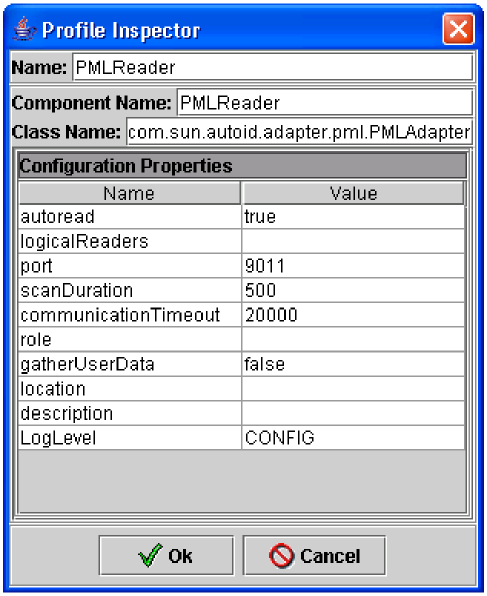
   The Select a Device Profile dialog box appears.



Intermec IF5 Readers
Zebra Printers
Printronix Printers
Matrics Readers
Feig ISCMR100 Readers
ThingMagic Mercury4 Readers
Alien Readers
ThingMagic Mercury3 Readers

4. **Select the device profile to edit and click Ok.**

   The Profile Inspector dialog box appears.



Name: PMLReader

Component Name: PMLReader

Class Name: com.sun.autoid.adapter.pml.PMLAdapter

Configuration Properties

| Name | Value |
|---|---|
| autoread | true |
| logicalReaders | |
| port | 9011 |
| scanDuration | 500 |
| communicationTimeout | 20000 |
| role | |
| gatherUserData | false |
| location | |
| description | |
| LogLevel | CONFIG |

The first field is the profile Name, which identifies the profile uniquely.

The next area of the dialog box shows the configuration properties. See Appendix A for detailed descriptions of these properties for each supported adapter.

5. **Type the new values, and click Ok.**

   At this point, your work is only saved to an in-memory database. You must save the results of your changes as described in the next step.

6. **Return to the RFID Configuration Manager window and choose File → Save.**

7. **Click Ok when prompted.**

# RFID Event Manager Component Overview

The primary types of RFID Event Manager components are filters and connectors. The RFID Software comes with a set of predefined filters and connectors that are used to create roles in the RFID system. Roles are defined in "Event Manager Concepts" on page 17.

## RFID Event Manager Filters

Zero or more filters may be defined in a role. Filters are linked from input events to other components that name them as outputs. The following properties define a filter:

- **name** – The unique name for the component.
- **classname** – The name of the Java class that implements the component.
- **properties** – Asequence of name/value pairs that are used for the configuration of the filter.
- **outputs** – A sequence of component names to be registered as event listeners to this component. The outputs normally designate one or more filters or connectors.

The supported filters are listed in the following table. For a description of the filter's associated properties, see the table listed in the Properties column.

**TABLE 2-5** RFID Event Manager Filters

| Name | Description | Properties |
|---|---|---|
| `BandPass` | Performs a pass filter on the reader EPCs. Events from readers that match the EPC mask are passed on to listeners, while other EPCs are not. | See TABLE B-4. |

**TABLE 2-5** RFID Event Manager Filters *(Continued)*

| Name | Description | Properties |
|---|---|---|
| Delta | Reports tags leaving and entering the radio frequency (RF) field. | See TABLE B-2. |
| EPC | Performs a pass filter on tag EPCs. EPCs that match the EPC mask are passed on to listeners, while other EPCs are not. | See TABLE B-3. |
| Smoothing | Creates a union of EPCs discovered over the number of specified *n* cycles. If an EPC was discovered in a cycle < *n*, the EPC is reported, if the EPC hasn't been seen in more than the last *n* cycles, the EPC is not reported. This filtering action is necessary because the RFID readers do not report tags with 100% tag accuracy. | See TABLE B-1. |

# RFID Event Manager Connectors

Zero or more connectors may be defined. Connectors are linked from input events to other components that name them as outputs. A connector is defined by the following properties:

- **name** – The unique name for the component.
- **classname** – The name of the Java class that implements the component.
- **properties –** A sequence of name/value pairs that are used for the configuration of the connector.

The supported connectors are described in the following table. For a description of the connector's associated properties, see the table listed in the Properties column.

**TABLE 2-6** RFID Event Manager Supported Connectors

| Name | Description | Properties |
|---|---|---|
| File Connector | Writes out PML core messages to a file. | See TABLE B-5. |
| HttpPml Connector | Writes out PML core messages to an HTTP connection. | See TABLE B-6. |
| JMS Connector | Writes out PML core messages to a JMS queue or topic. | See TABLE B-7. |
| Socket Connector | Creates a socket connection and starts writing PML core messages to the connection. | See TABLE B-9 |
| ServerSocket Connector | Creates a server socket and listens for a connection. This connector starts writing PML core messages once the connection is established. | See TABLE B-11 |

## Custom RFID Filter and Connectors

The RFID Software supports the creation of customized filters and connectors. Use the Sun Java System Software Toolkit 3.0 to create custom components. See the *Sun Java System RFID Software 3.0 Developer's Guide* for details.

CHAPTER 3

# Distributed RFID Event Manager Environment

This chapter includes the following topics:

- General Distributed Installation Considerations
- Managing a Distributed Installation

## General Distributed Installation Considerations

RFID Event Manager Execution Agents communicate and gather information from the RFID readers. A single Execution Agent is capable of controlling and collecting information from multiple RFID Readers.

In a large deployment, you can install a copy of the Execution Agent on multiple hosts in the network. Each Execution Agent can then manage and communicate with a subset of the RFID Readers.

Which readers communicate with a particular Execution Agent is specified by the RFID Event Manager. Deploying multiple Execution Agents enhances the availability of the system because each Execution Agent is capable of carrying out the workload of other Execution Agents if a particular Execution Agent becomes unavailable.

You use the RFID Configuration Manager to define Configuration Objects that encapsulate how to process the data received from the readers. The Control Station provisions the Configuration Objects to the Execution Agents as units of work.

Execution Agents and their Control Station need to share a common Jini Group in order to communicate with each other. At installation time, you are asked to define a group name for your deployment. This name needs to be the same across the set of multiple Execution Agents you want to communicate with a single Control Station.

You can change the Control Station's group name after installation by invoking the RFID Configuration Manager. The only way to change the group name for the Execution Agents is to reinstall them, so be careful when you define the group name.

**Note –** All of your deployment information is managed from the Control Station. No configuration information is lost by reinstalling an Execution Agent.

# Managing a Distributed Installation

Use a custom installation to create distributed installations where the Execution Agent components are located on separate machines that are remote to the Control Station machine. See the *Sun Java System RFID Software 3.0 Installation Guide* for more information on creating a distributed installation for your RFID Software.

## Distributed Installation Component Scripts

The RFID Software includes scripts that are designed to run or stop various software components. When a distributed installation is created, the Control Station and Execution Agents can be started separately by invoking their individual startup scripts. The same is true for separately stopping the components. These scripts also include a “restart” option for use with their respective components.

**Note –** The Control Station must be started before the Execution Agent can be started. Starting the Execution Agent when no Control Station is present results in an error. You need to stop and restart the Execution Agent after the Control Station has been started.

### Starting and Stopping the Control Station

If the Control Station components are installed as part of a Custom installation, use the script `station` stored in /*rfid-install-dir*/`bin` to start the Control Station.

To use the script, change to the specified directory and type one of the following:

- **To start** the Control Station, type **`station start`**.
- **To stop** the Control Station, type **`station stop`**.
- **To show the status** of the Control Station, type **`station status`**.
- **To show the version** of the Control Station, **`station version`**.

## Starting and Stopping the Execution Agent

Similarly, on hosts where the Execution Agent components are installed without the Control Station, you can find the script `agent` installed in /*rfid-install-dir*/`bin`.

To use the script, change to the specified directory and type one of the following, as needed:

- **To start** the Execution Agent, type **`agent start`**.
- **To stop** the Execution Agent, **`agent stop`**.
- **To show the status** of the Execution Agent, type **`agent status`**.

CHAPTER 4

# Configuring Communication With SAP Auto-ID Infrastructure

Sun Java System RFID Software 3.0 (RFID Software) includes a plug-in to enable integration with the SAP Auto-ID Infrastructure 2.1 (SAP AII) component of the SAP NetWeaver® platform. SAP AII enables communication between the SAP® R/3® software, the mySAP™ Supply Chain Management solution (mySAP SCM solution), and other SAP Enterprise Resource Planning (ERP) components using device management/device controller software. The RFID Event Manager plays the role of a device manager in the SAP architecture.

This chapter includes the following topics:

- Configuration Overview for the Plug-In for SAP AII
- Configuring the RFID Event Manager to Communicate With SAP AII Software
- Configuring SAP AII to Communicate With the RFID Event Manager

## Configuration Overview for the Plug-In for SAP AII

To configure the connection between the RFID Software and SAP AII, you must complete configuration procedures for both the RFID Software and SAP AII. The procedures for the RFID Software use the RFID Configuration Manager, which is a component of the RFID Event Manager. Therefore, you must install the RFID Event Manager and the plug-in for SAP AII before performing any procedures in this chapter. See the *Sun Java System RFID Software 3.0 Installation Guide* if you have not installed the RFID Event Manager. Perform a custom installation so that you can install the plug-in for the SAP AII software.

Use the RFID Configuration Manager to configure your RFID device properties. If you are not familiar with this configuration tool, review Chapter 2 before performing the procedures in this chapter.

Setting up communication between the RFID Event Manager and the SAP AII software also requires familiarity with the SAP software. You must also have SAP AII installed to complete the procedures in this chapter. This guide does not cover the details of SAP AII installation or configuration. However, the intent is to provide sufficient information to enable a user with reasonable familiarity with SAP AII to complete the necessary tasks.

**Note –** Configure the RFID Event Manager first, then configure SAP AII. You must perform all the procedures in this chapter in the order in which they appear.

# Architecture Overview of the RFID Software Plug-In for SAP AII

The RFID Event Manager performs the role of a device manager in the SAP AII architecture. The SAP AII software packages *Tag Commissioning* requests in XML messages that are sent to the device manager. XML messages are expected back from the device manager when a business event occurs. The plug-in for SAP AII consists of a web service and a connector. The web service is implemented by the RFID Event Manager and hosted on the application server. The connector is the SAPLogger connector.

The SAP AII plug-in's web service has the following characteristics:

- The web service listens for XML messages conforming to the SAP AII-DC (fixed reader) protocol. The protocol specifies the format of the Tag Commissioning commands. SAP AII generates these XML Tag Commissioning messages to program the RFID tag and print the RFID label.
- The web service receives the HTTP message, programs the tag, prints the label, and returns `HTTP OK` when the print command is successful.

The SAP AII connector (SAPLogger) has the following characteristics:

- The SAPLogger is installed at the end of a reader's Business Processing Semantics (BPS) chain. For a description of the BPS concept, see "Event Manager Concepts" on page 17.
- The SAPLogger indicates the status of RFID tags discovered by using HTTP to post notifications to SAP AII. These status codes are defined in SAP AII as business events that trigger business processes.
- The SAP AII device ID and the business event code used with the connector's notifications are properties that are associated with each reader device.

- The SAPLogger connector interprets these properties and maps them to a specific XML message sent back to SAP AII.
- The connector's properties are set by using the RFID Configuration Manager. See "Configuring the RFID Event Manager to Communicate With SAP AII Software" on page 64.

---

**Note –** The plug-in's SAP AII web service is only used for incoming requests. The plug-in's SAP AII connector is only used for outgoing requests.

---

The following illustration shows an example where a Tag Commissioning XML message is sent from SAP AII. The plug-in's web service in the RFID Event Manager receives the message, determines the reader ID, and discovers the device. The device then uses the information in the message to program the tag and print the label. When a tag is read by an RFID reader, an HTTP business event message is generated by the plug-in's RFID Event Manager SAP AII connector. The message includes the ID of the reader that generated the business event and the RFID tag information. The message is sent to a URL where SAP AII listens for notifications.

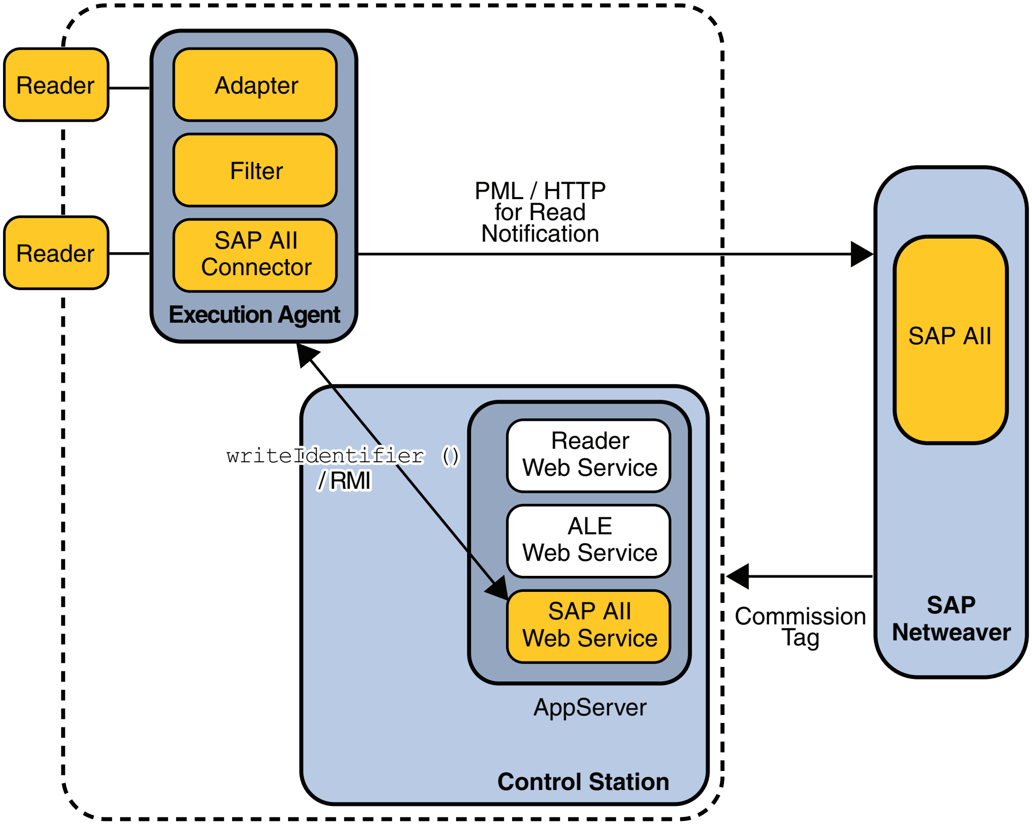


**FIGURE 4-1** Communication Flow Between the RFID Event Manager and SAP AII.

# Configuring the RFID Event Manager to Communicate With SAP AII Software

You need to do the following to configure the RFID Event Manager so that it can communicate with SAP AII:

- Define your RFID devices.
- Configure the parameters for the SAP AII reader role (work flow).
- Define the configuration objects for communication with SAP AII.

## ▼ To Define the RFID Devices

1. **If you have not already done so, start the RFID Configuration Manager.**

   See "To Start the RFID Configuration Manager (UNIX)" on page 22 or "To Start the RFID Configuration Manager (Microsoft Windows)" on page 24.

   The RFID Configuration Manager appears.

2. **Remove the `Demo` role from the drawing pane by following these steps:**

   a. **Click on the `Demo` configuration object in the drawing pane.**

      This action selects the object. There is no visual cue to indicate that the role is selected.

   b. **From the main menu, choose Configuration → Delete.**

      A confirmation dialog box appears and displays the name of the configuration object that is selected.

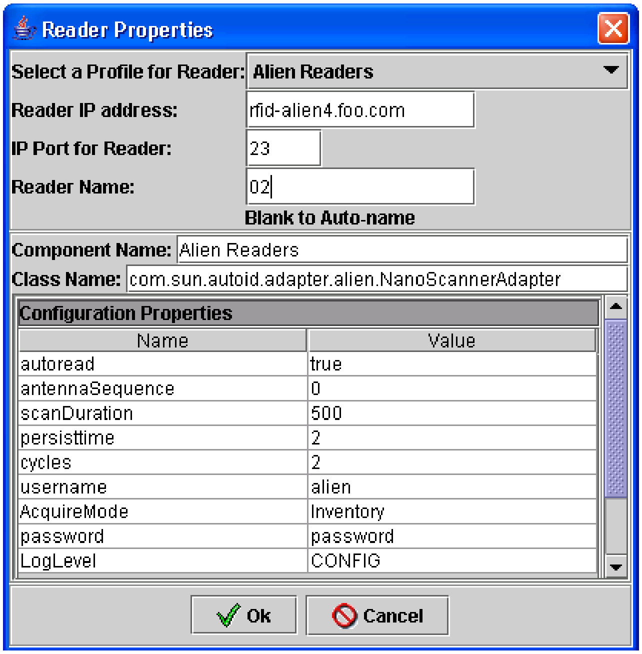
   c. **Confirm that the `Demo` object is named in the confirmation message and click Ok.**

3. **Choose Devices → New**.

   The Reader Properties dialog box appears.

4. **Click Please Select a Profile to see a drop-down list of possible device profiles.**

   Select the profile for your reader manufacturer and model. A profile might apply to more than one model of reader. The following example shows the profile for the Alien Reader.



Select a Profile for Reader: Alien Readers

Reader IP address: rfid-alien4.foo.com

IP Port for Reader: 23

Reader Name: 02

Blank to Auto-name

Component Name: Alien Readers

Class Name: com.sun.autoid.adapter.alien.NanoScannerAdapter

Configuration Properties

| Name | Value |
|---|---|
| autoread | true |
| antennaSequence | 0 |
| scanDuration | 500 |
| persisttime | 2 |
| cycles | 2 |
| username | alien |
| AcquireMode | Inventory |
| password | password |
| LogLevel | CONFIG |

5. **In the Reader IP address field, type the IP address for the specific device.**

6. **In the IP Port for Reader field, type the port number for the device.**

7. **In the Reader Name field, type the name for the device.**

   Select a name that you want SAP AII to use to communicate with this device. If no name is defined, the RFID Configuration Manager generates a name based on the profile type, the host name, and the port number. This name should match the name used on the Auto-ID Master Data: RFID Device configuration screen in SAP AII.

8. **(For Printers Only) To define a printer, you specify the appropriate RFID tag formats to be used by the printer by using the `template` property.**

   - The template property name is of the form `template.`*f1*, `template.`*f2*, and so on.
   - The `template` property value is of the form `http://localhost:52493/system/`*template-file*. Replace the variable, *template-file*, with the file name containing the template definition.
   - The `template` property specifies the default format for the printer.

   A printer template specifies the placement of the text (the format) on the RFID tag and is customized for the type of data that is being sent to the printer and for the specific RFID tag printer model. A printer might have different formats that correspond to different RFID tag functions. Each print format needs its own `template` property. For example, the format of the text on an RFID tag printed for a specific inventory unit (a box) might differ from the format of the RFID tag printed for a pallet containing multiple inventory units. You can specify multiple formats for each printer. To do this add a new template property for each print format that you need. The print format defined by the original template property is the default template used by the printer. Each additional template property should be named as follows, template.f1, template.f2, and so on. The value for each template (template, template.f1, template.f2 and so on) property specifies the text file that contains the print format.

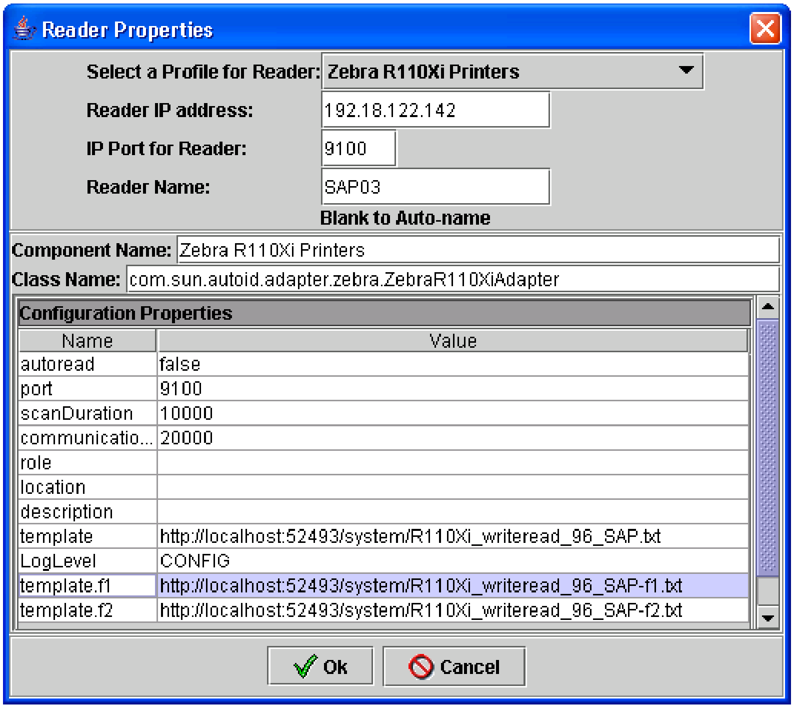
- You need to manually create and place the template files in the following directory corresponding to your platform:
- **Solaris OS** – /etc/opt/SUNWrfid/system
- **Linux** – /etc/opt/sun/rfidem/system
- **Windows** – C:\Program Files\Sun\RFID Software\rfidem\config\system

---

**Note –** As part of the printer definition, you must specify the printer template in SAP AII by using the Tag Commissioning operation. When you do the SAP AII configuration, define the printer format by specifying only the name of the template *file*. Do not specify the entire URL. The name of the template in the SAP AII configuration corresponds to the variable portion of the template property, *f1*, *f2* and so on.

---

The following screen capture shows a printer with the default printer template and two additional printer formats as specified by the properties, template.f1 and template.f2. Each template property has a different file name specified in the value field.



Select a Profile for Reader: Zebra R110Xi Printers

Reader IP address: 192.18.122.142

IP Port for Reader: 9100

Reader Name: SAP03

Blank to Auto-name

Component Name: Zebra R110Xi Printers

Class Name: com.sun.autoid.adapter.zebra.ZebraR110XiAdapter

Configuration Properties

| Name | Value |
|---|---|
| autoread | false |
| port | 9100 |
| scanDuration | 10000 |
| communicatio... | 20000 |
| role | |
| location | |
| description | |
| template | http://localhost:52493/system/R110Xi_writeread_96_SAP.txt |
| LogLevel | CONFIG |
| template.f1 | http://localhost:52493/system/R110Xi_writeread_96_SAP-f1.txt |
| template.f2 | http://localhost:52493/system/R110Xi_writeread_96_SAP-f2.txt |

9. **When you are done defining the device properties, click Ok.**

10. **Repeat the steps 3 through 8 to define all your readers and printers.**

11. **When you have finished defining your devices, proceed to the next section and define the RFID Event Manager's SAP AII connection properties.**

## Defining the RFID Event Manager's SAP AII Connection Properties

The RFID Software provides two default Roles for communication with SAP AII. The roles are the following:

- SAP AII printer role
- SAP AII reader role

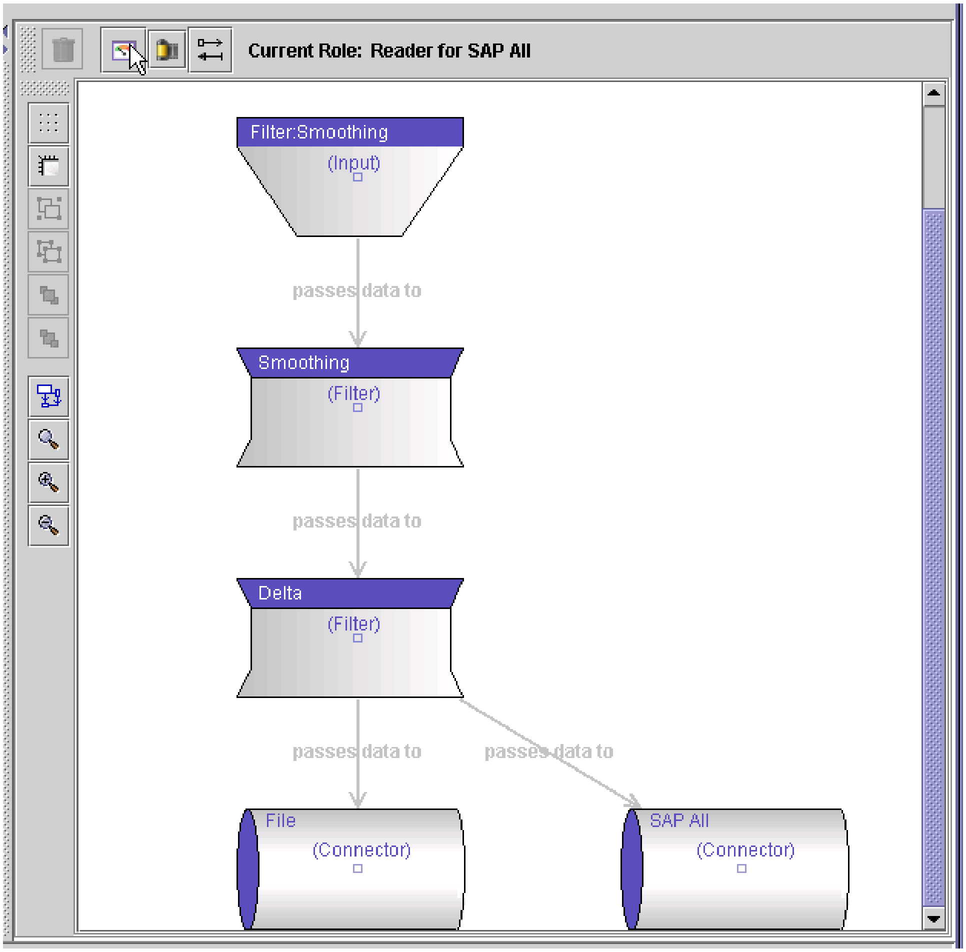
The printer role is used to connect printer devices that receive one-way requests from the software. This role does not require any configuration.

The reader role is used to send notifications to SAP AII by using the SAP AII-DC 1.0 protocol. In the SAP AII reader role, you specify the connection parameters (host name and port number) for SAP AII. By default, the RFID Event Manager defines the controller device ID as `SunEventManager`. You can specify a different controller device ID by updating the properties of the SAP AII connector in the role.

## ▼ To Define the RFID Event Manager's SAP AII Connection Parameters

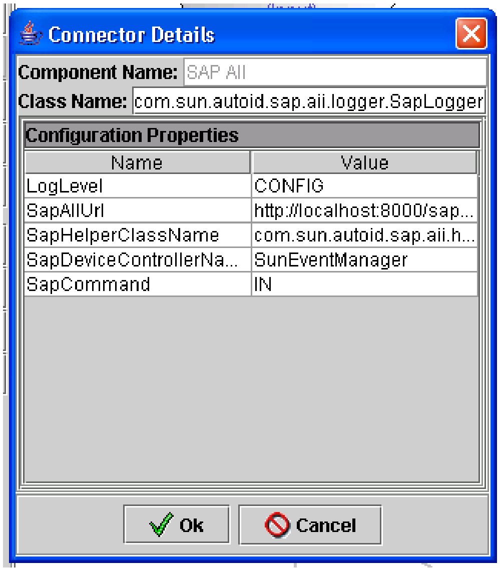
1. **From the RFID Configuration Manager navigation tree (in the left pane), expand the Roles node and double-click the Reader for SAP AII role.**

   The components of the role appear in the drawing pane of the editor as shown in the following screen capture.



2. **Select and right-click the Connector for SAP AII.**
3. **Choose Edit from the contextual menu.**

   The Connector Details dialog box appears.



Component Name: SAP AII

Class Name: com.sun.autoid.sap.aii.logger.SapLogger

Configuration Properties

| Name | Value |
|---|---|
| LogLevel | CONFIG |
| SapAIIUrl | http://localhost:8000/sap... |
| SapHelperClassName | com.sun.autoid.sap.aii.h... |
| SapDeviceControllerNa... | SunEventManager |
| SapCommand | IN |

4. **Update the `SAPAIIUrl` property field with the URL where your SAP AII software listens for notifications.**

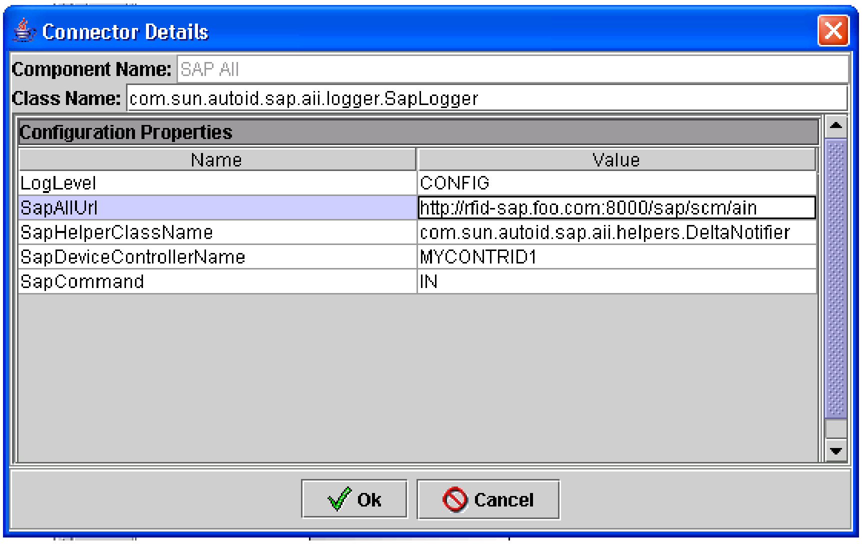
   Replace the default URL with your specific configuration details.

5. **Confirm the value of the `SapDeviceControllerName` property.**

   The `SapDeviceControllerName` property specifies the device controller ID used in notifications to SAP AII. You must use this value as the device controller ID in the SAP AII Auto-ID Master Data: RFID Device configuration screen.

   The following screen capture shows the connection settings for SAP AII software running on host `rfid-sap.foo.com` , which is listening on port 8000 with path `/sap/scm/ain`.

   For illustration purposes, the device controller name has also been modified to `MYCONTRID1`. There's no need to change it from the default if you use `SunEventManager` as the name of your device controller ID in the Auto-ID Master Data: RFID Device configuration screen.



Component Name: SAP AII

Class Name: com.sun.autoid.sap.aii.logger.SapLogger

Configuration Properties

| Name | Value |
|---|---|
| LogLevel | CONFIG |
| SapAIIUrl | http://rfid-sap.foo.com:8000/sap/scm/ain |
| SapHelperClassName | com.sun.autoid.sap.aii.helpers.DeltaNotifier |
| SapDeviceControllerName | MYCONTRID1 |
| SapCommand | IN |

6. **Confirm the value of the `SapCommand` property.**

   Every XML notification message to SAP AII includes a command element. SAP AII provides the ability to define new commands and map them to business rules. The `SapCommand` property defines the default command that is to be included in the notification to SAP AII. Because this command is likely to be different for readers in different business roles, you can change the command for each configuration object. Changing the `SapCommand` property is described in the procedure "To Define the Reader Configuration Object for SAP AII" on page 74.

7. **Click Ok.**

8. **Continue to the next section and define your configuration objects.**

## Defining Configuration Objects for SAP AII Communication

Each configuration object represents a specific device's work flow for communication with SAP AII. A configuration object is an implementation of a role that uses a defined reader or printer.

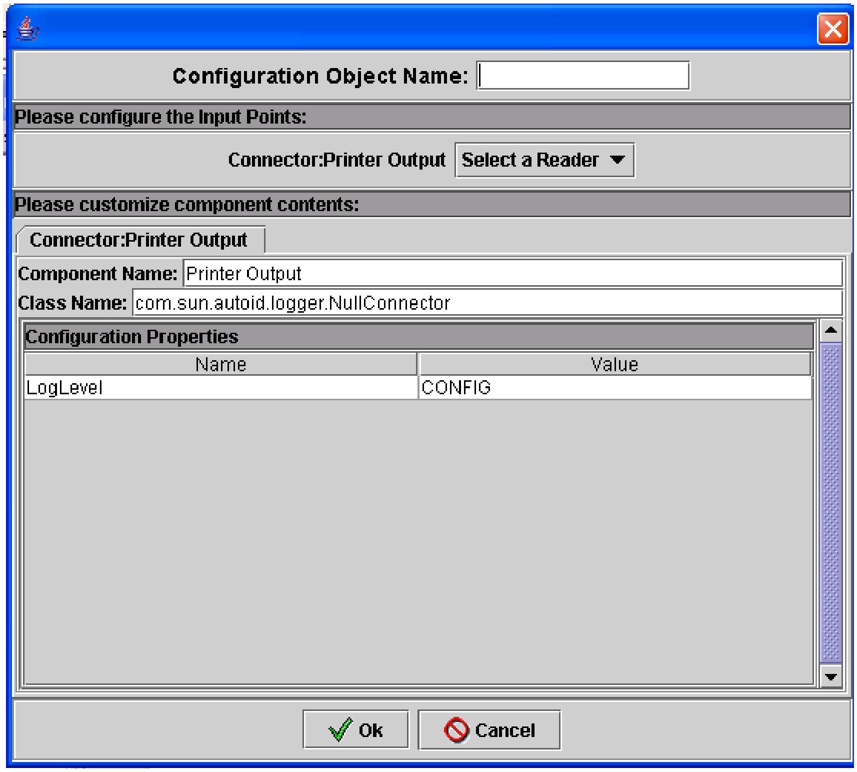
## ▼ To Define the Printer Configuration Object for SAP AII

1. **In the RFID Configuration Manager, choose Configuration → New.**

   The Select the Base Role dialog box appears and lists the available roles.

2. **Select Printer for SAP AII and click Ok.**

   The Configuration Object dialog box appears.



Configuration Object Name:

Please configure the Input Points:

Connector:Printer Output Select a Reader

Please customize component contents:

Connector:Printer Output

Component Name: Printer Output

Class Name: com.sun.autoid.logger.NullConnector

Configuration Properties

| Name | Value |
|---|---|
| LogLevel | CONFIG |

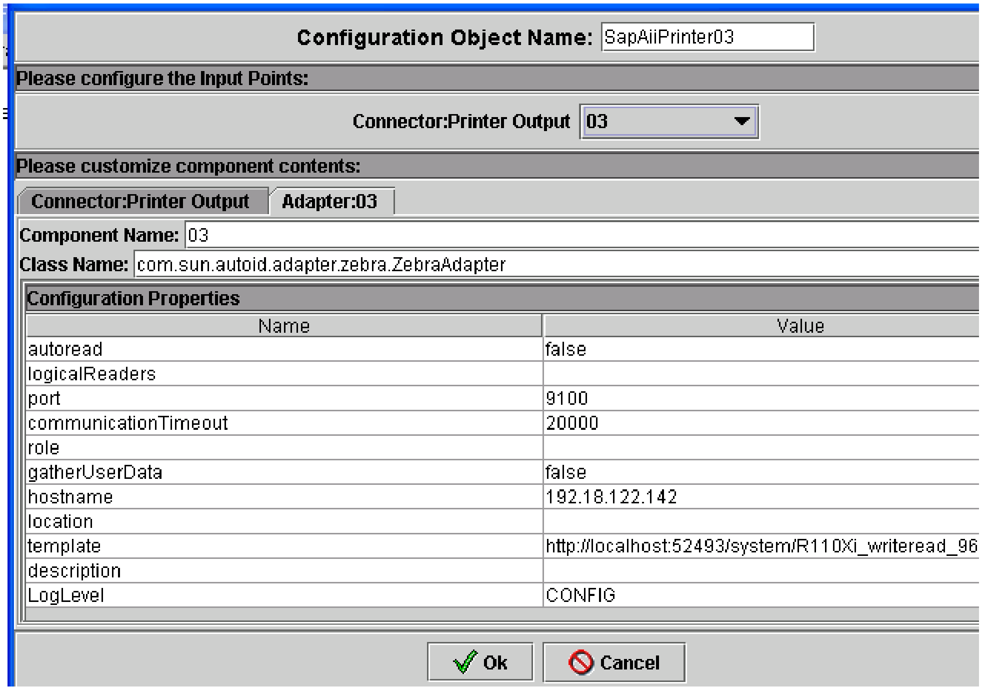
3. **In the Configuration Object Name field, type a unique name for the configuration.**

   For example: `SapAiiPrinter03`

4. **In the section Please configure the Input Points, click Select a Reader.**

5. **Select your printer device from the drop-down list of available devices.**

   The following example shows the printer named `03`, which was previously defined.



Configuration Object Name: SapAiiPrinter03

Please configure the Input Points:

Connector:Printer Output: 03

Please customize component contents:

Component Name: 03

Class Name: com.sun.autoid.adapter.zebra.ZebraAdapter

Configuration Properties

| Name | Value |
|---|---|
| autoread | false |
| logicalReaders | |
| port | 9100 |
| communicationTimeout | 20000 |
| role | |
| gatherUserData | false |
| hostname | 192.18.122.142 |
| location | |
| template | http://localhost:52493/system/R110Xi_writeread_96 |
| description | |
| LogLevel | CONFIG |

6. **Confirm the values of the properties, and click Ok.**

## ▼ To Define the Reader Configuration Object for SAP AII

1. **In the RFID Configuration Manager, choose Configuration → New.**

   The Select the Base Role dialog box appears and lists the available roles.

2. **Select Reader for SAP AII and click Ok.**

   The Configuration Object dialog box appears.

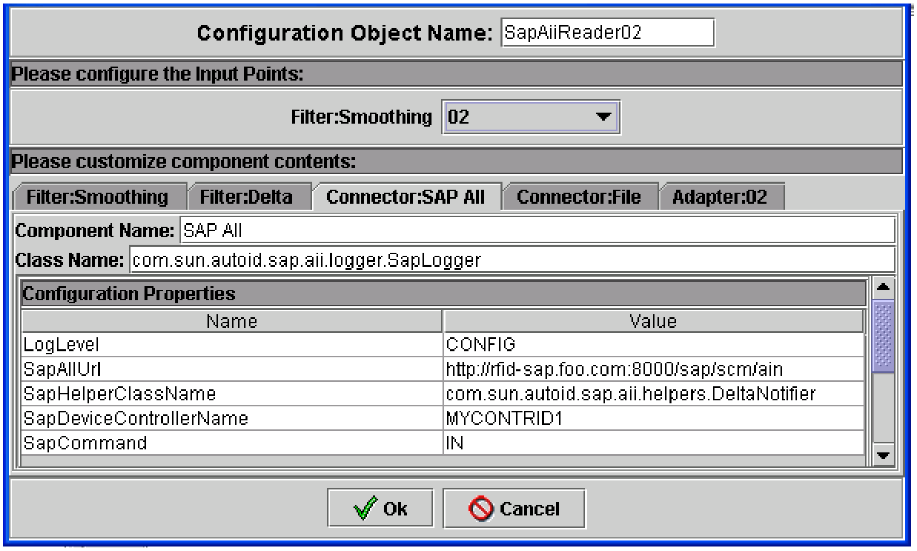
3. **In the Configuration Object Name field, type a unique name for the configuration.**

   For example: `SapAiiReader02`

4. **In the section labeled 'Please configure the Input Points:', click Select a Reader.**

5. **Select the reader from the drop-down list of available devices.**

   The following example shows the reader named `02,` that was previously defined.



Configuration Object Name: SapAiiReader02

Please configure the Input Points:

Filter:Smoothing 02

Please customize component contents:

Component Name: SAP AII

Class Name: com.sun.autoid.sap.aii.logger.SapLogger

Configuration Properties

| Name | Value |
|---|---|
| LogLevel | CONFIG |
| SapAIIUrl | http://rfid-sap.foo.com:8000/sap/scm/ain |
| SapHelperClassName | com.sun.autoid.sap.aii.helpers.DeltaNotifier |
| SapDeviceControllerName | MYCONTRID1 |
| SapCommand | IN |

6. **Select the tab Connector:SAPAII to review the values of the Configuration Properties.**

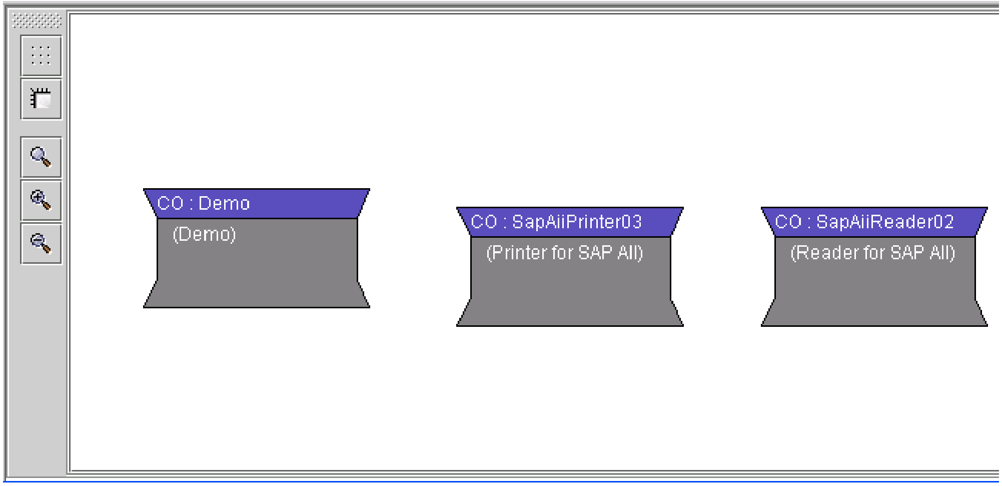
   You can modify these properties without affecting other reader configurations.

   By default, the `SapCommand` property contains the value `IN` that tells the device to send the notification command, `IN`, to SAP AII. The values for the `SapAIIUrl` and `SapDeviceControllerName` properties reflect the values that were used during the role configuration.

7. **Click Ok.**

8. **Save your work by choosing File → Save and click Ok when prompted.**

   You now have two device configurations as shown in the following screen capture. Repeat the procedure to add more devices.



# Configuring SAP AII to Communicate With the RFID Event Manager

After configuring the RFID Event Manager, you must configure the SAP AII side of the connection. You use the SAP AII software to perform this procedure. This procedure does not describe installation of the SAP AII software.

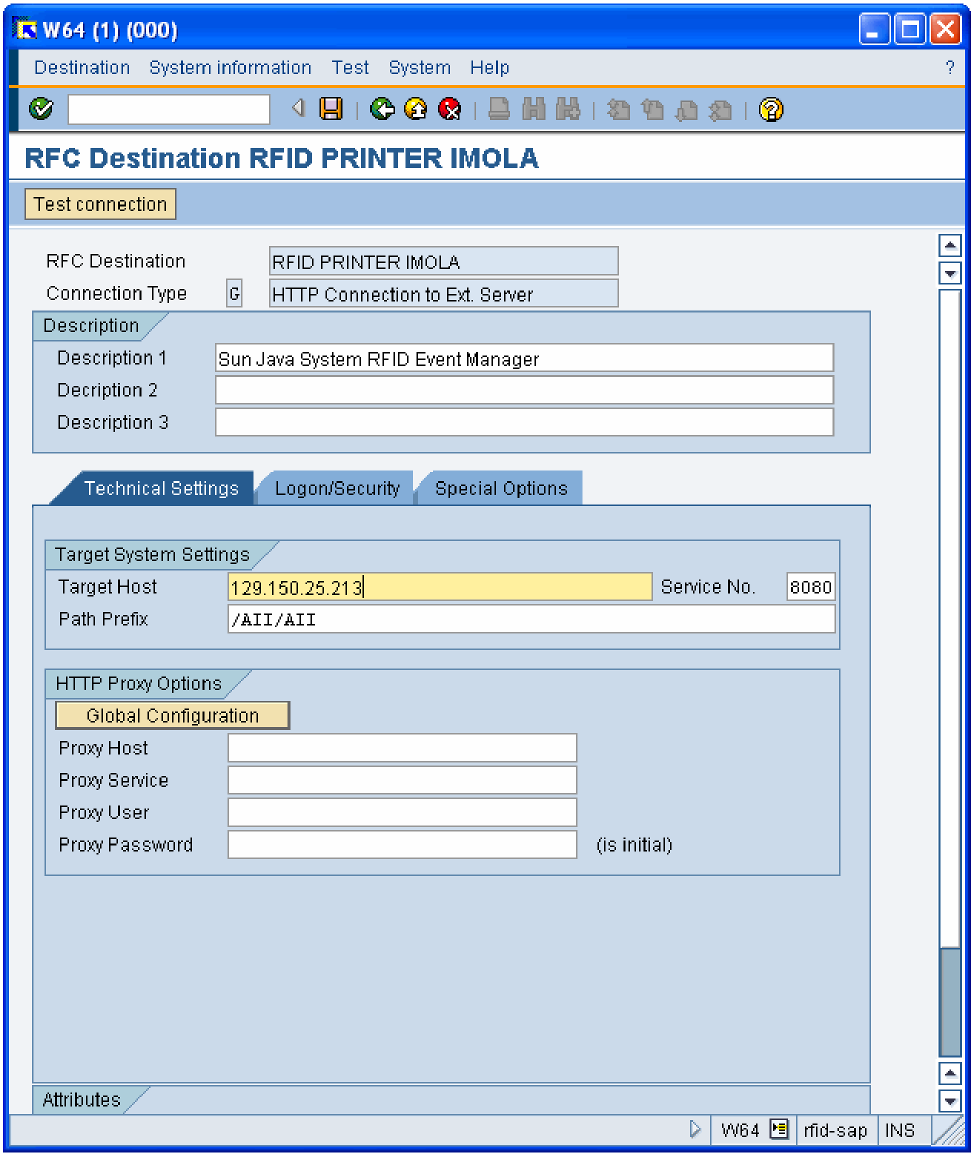
**Prerequisites** – During this procedure, you will test the connection between the RFID Event Manager and SAP AII, so you need to complete these prerequisites:

- If you have not already done so, perform the RFID Event Manager configuration. See “Configuring the RFID Event Manager to Communicate With SAP AII Software” on page 64
- If you have not already done so, stop and restart the RFID Event Manager to enable your RFID Event Manager configuration changes to take effect. See “To Start the RFID Event Manager” on page 45.
- Start the application server.

## ▼ To Configure SAP AII Communication With the RFID Event Manager

1. **Log in to your SAP AII installation.**
2. **Execute transaction** `/nsm59` **to enable use of the SAP AII interface's Display and Maintain RFC Destinations feature.**
3. **Define your RFC as described in the SAP documentation.**

   The following SAP AII screen capture shows the configuration for the RFID Event Manager deployed at IP address: `129.150.25.213`, listening on port 8080, at default path `/AII/AII`.



RFC Destination RFID PRINTER IMOLA

RFC Destination: RFID PRINTER IMOLA

Connection Type: G HTTP Connection to Ext. Server

Description

Description 1: Sun Java System RFID Event Manager

Decription 2:

Description 3:

Target System Settings

Target Host: 129.150.25.213 Service No. 8080

Path Prefix: /AII/AII

HTTP Proxy Options

Proxy Host:

Proxy Service:

Proxy User:

Proxy Password: (is initial)

Attributes

4. **Create or modify (as needed) a new RFC destination with a connection type of HTTP Connections to Ext. Server.**

5. **Click Test Connection to test the communication with the RFID Event Manager.**

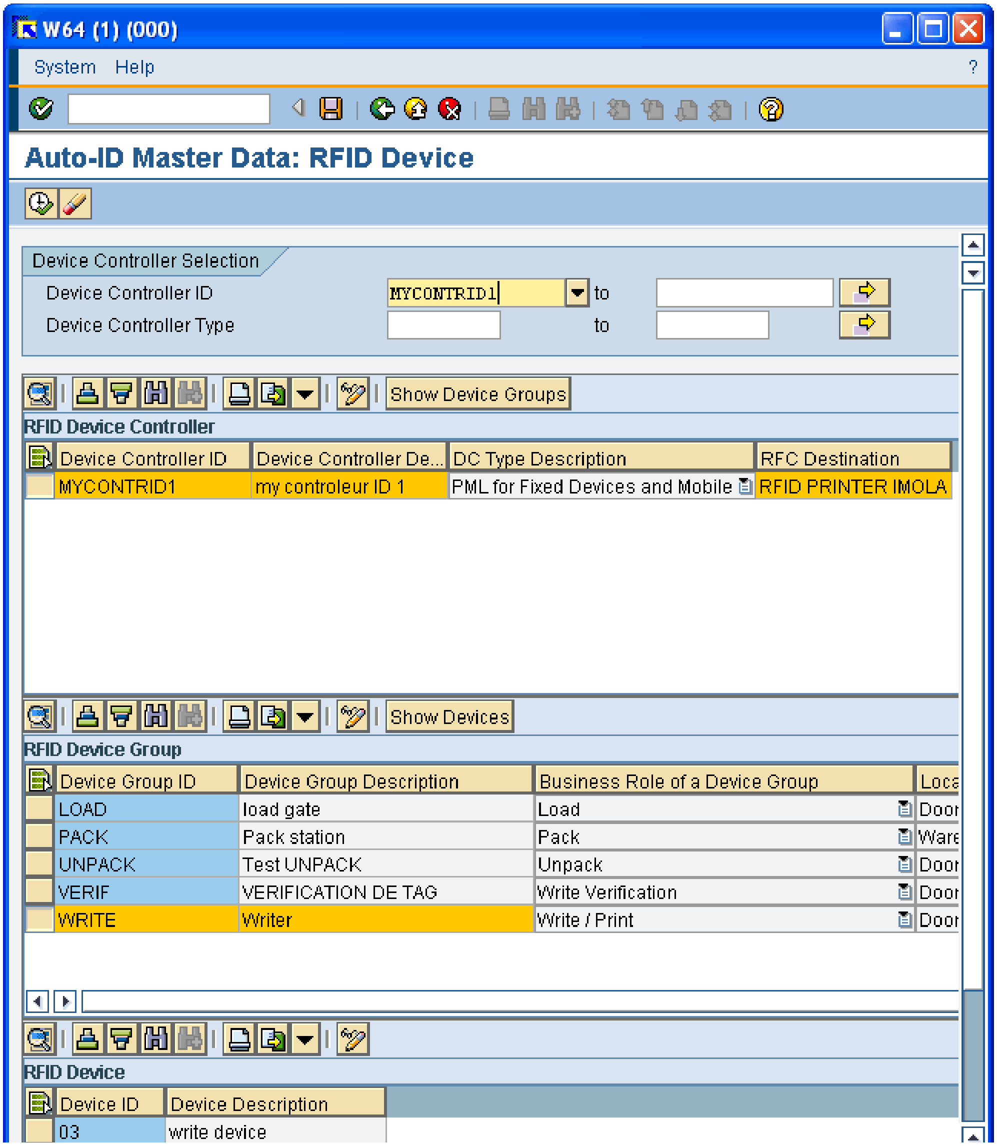
   You should receive the error: `HTPP/1.1 500 Internal Server Error` - this is expected. This error indicates that although the RFID Event Manager was able to receive the test command, because the command does not follow the SAP AII-DC 1.0 protocol, it cannot be parsed. This is the expected result at this point.

   If you received a different error, confirm the following:

   - The RFID Event Manager was installed successfully using the custom installation option to install the plug-in for SAP AII. See the *Sun Java System RFID Software 3.0 Installation Guide*.
   - The RFID Event Manager was started successfully. See "Starting the RFID Configuration Manager" on page 22.
   - The target system settings were entered correctly:
     - The IP address is correct.
     - The port number is correct.
     - The path is correct.

6. **Configure the RFID Event Manager as a device controller by using the SAP AII Auto-ID Master Data: RFID Device configuration screen.**

   It is critical that the SAP AII Device Controller ID matches the value of the Sun RFID Event Manager property, `SAPDeviceControllerName`. The SAP AII Device ID definitions must also match the name or EPC of the devices that are defined in the RFID Event Manager. These data fields (Device Controller ID and Device ID definitions) are shown in the following screen capture.



Auto-ID Master Data: RFID Device

Device Controller Selection

| | | | | |
|---|---|---|---|---|
| Device Controller ID | MYCONTRID1 | to | | |
| Device Controller Type | | to | | |

RFID Device Controller

| Device Controller ID | Device Controller De... | DC Type Description | RFC Destination |
|---|---|---|---|
| MYCONTRID1 | my controleur ID 1 | PML for Fixed Devices and Mobile | RFID PRINTER IMOLA |

RFID Device Group

| Device Group ID | Device Group Description | Business Role of a Device Group | Loca |
|---|---|---|---|
| LOAD | load gate | Load | Door |
| PACK | Pack station | Pack | Ware |
| UNPACK | Test UNPACK | Unpack | Door |
| VERIF | VERIFICATION DE TAG | Write Verification | Door |
| WRITE | Writer | Write / Print | Door |

RFID Device

| Device ID | Device Description |
|---|---|
| 03 | write device |

Copyright SAP AG 2006. All rights reserved.

CHAPTER 5

# Monitoring the RFID System

The RFID Management Console is a web-based application for monitoring the readers and components of the RFID Event Manager system.

This chapter includes the following topics:

- RFID Management Console Overview
- Performing RFID Management Console Administration Tasks
- Performing RFID Reader Network Management Tasks
- Monitoring the Status of Your RFID Reader Network By Using Alerts

## RFID Management Console Overview

The RFID Management Console is a web-based application that can be accessed using a web browser to both monitor and modify the various components of the RFID Event Manager. The RFID Management Console shows the status of the readers and components and can be used to quickly assess the overall state of the RFID Event Manager system.

The RFID Management Console may also be used to modify various read and write properties of the RFID Event Manager components as the system is running.

The Sun Java System RFID Software 3.0 release has enhanced the RFID Management Console in the area of user creation and management. The previous release provided one user, `admin`, of the console. RFID Software 3.0 enables the creation of multiple role-based user logins. By default, the system still defines the `admin` user. The `admin` user is assigned the following administrative privileges:

- View all readers
- Create new users and groups
- Define privileges for users

For more details, see "Creating and Managing RFID Management Console Users" on page 84.

## ▼ To Access the RFID Management Console

1. **Confirm that the RFID Event Manager and your application server are running.**

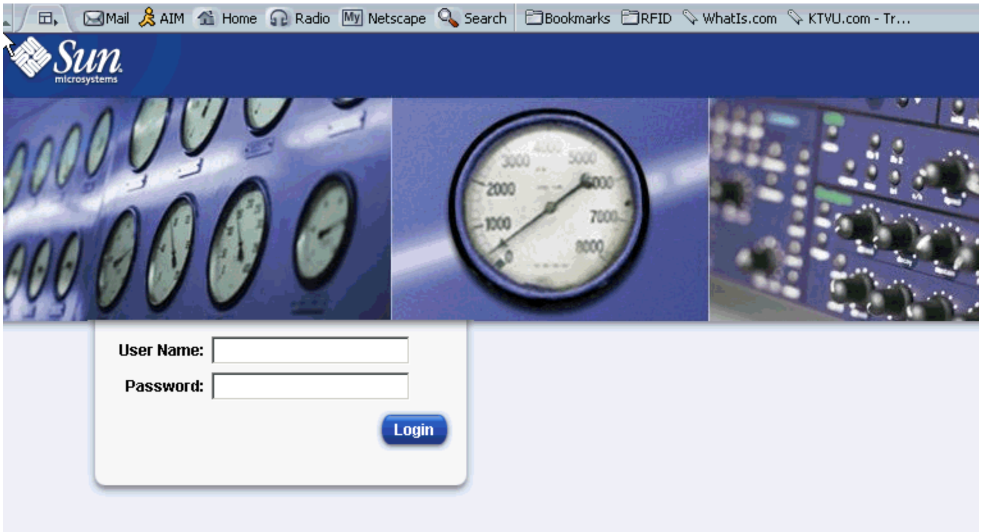
   To use the RFID Management Console, your application server must be running. To see all of the screens and data for the RFID Management Console, the RFID Event Manager must also be running.

2. **Open a supported web browser and type the RFID Management Console URL.**

   The URL is of the following format: `http://`*hostname*:*port-number*`/sdui`. For example, `http://myhost:8080/sdui`.

   - The variable, *hostname*, is the name of the system where the RFID Management Console is deployed. You can also use the IP address of the machine, such as `10.6.165.71`.
   - The variable, *port-number*, is the HTTP port number for your installation of Sun Java System Application Server 8.1.

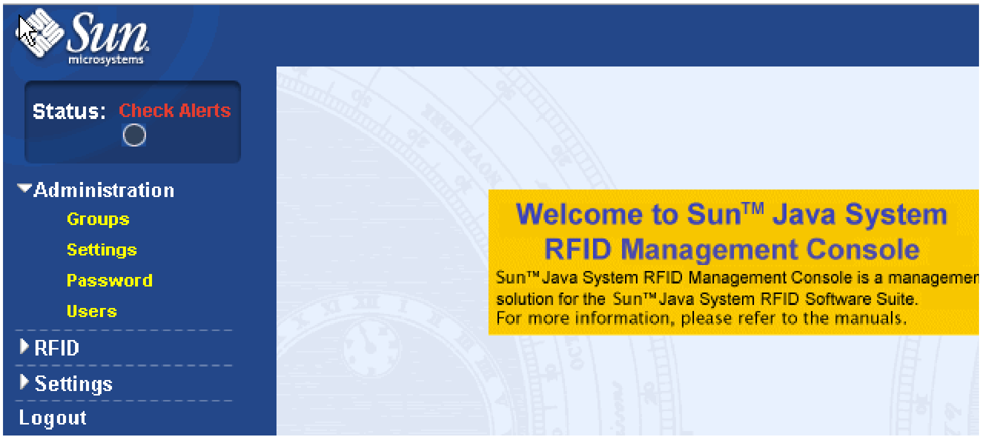
   The RFID Management Console login screen appears.



3. **Type the user name and password.**

   - The default user name is `admin`.
   - The default password is `admin`.

   The RFID Management Console Welcome screen appears.

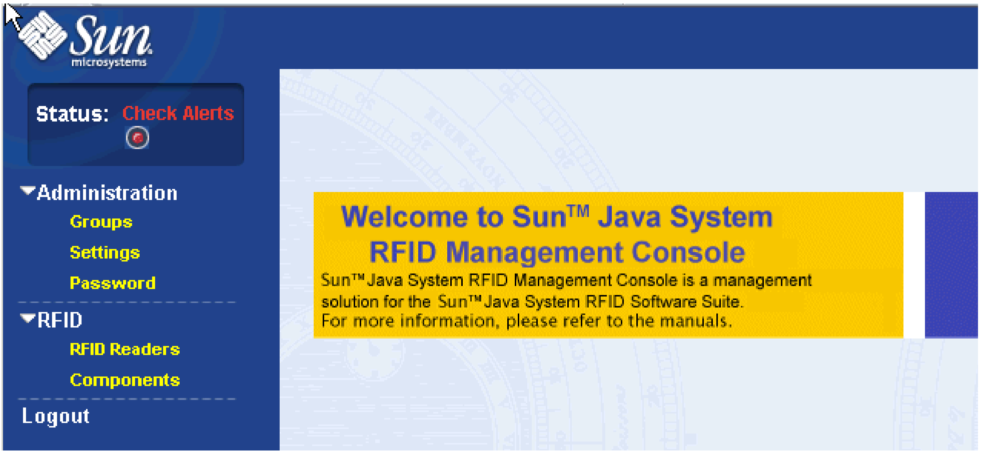


Welcome to Sun™ Java System RFID Management Console

Sun™ Java System RFID Management Console is a managemer solution for the Sun™ Java System RFID Software Suite. For more information, please refer to the manuals.

4. **Choose the task to perform from the links on the left side of the screen.**

   See the following sections of this chapter for more information on using the RFID Management Console.



Welcome to Sun™ Java System RFID Management Console

Sun™ Java System RFID Management Console is a management solution for the Sun™ Java System RFID Software Suite. For more information, please refer to the manuals.

# Performing RFID Management Console Administration Tasks

From the RFID Management Console Administration menu, you can perform the following tasks:

- Manage groups.
- Manage settings.
- Change passwords.
- Manage users.

## Creating and Managing RFID Management Console Users

When you log in to the RFID Management Console for the first time, you must log in as the default `admin` user. The `admin` user has *administrator privileges* and can view all readers and groups. The `admin` user also creates new users and groups. New users can have administrator privileges. A user with administrator privileges is called an administrator and has the same privileges as the default `admin` user. An administrator can view all groups and readers.

Administrator privileges enable the user to create new users and groups. By default, any user created with administrator privileges is automatically assigned to the `All Readers` group. This group enables the user to view all readers for potential assignment to the various reader groups. An administrator can also assign Add/Delete Reader privileges to a user. These privileges enable the user to add and delete readers from owned groups. Otherwise, the user can only view the list of readers that an administrator has assigned to its groups.

**Note –** An administrator automatically receives Add/Delete Reader privileges.

An administrator can also create new groups and assign specific readers to the groups as they are created. This privilege enables an administrator to control the readers that are viewed by each user.

A user without administrator privileges (a nonadministrator) cannot create new users or new groups. A nonadministrator can be assigned Add/Delete Reader privileges and can then assign readers to his or her own groups.

In summary, the administrator can define new users as defined in the following table.

**TABLE 5-1** RFID Management Console User Roles and Privileges

| Action | Administrator | Nonadministrator Without Reader Privileges | Nonadministrator With Reader Privileges |
|---|---|---|---|
| Create users | X | | |
| Create groups | X | | |
| Assign users to groups | X | | |
| Assign readers to groups | X | | X |
| View groups | Can view all groups | Can only view groups to which the user belongs | Can only view groups to which the user belongs |
| View readers | Can view all readers | Can only view readers in the groups to which it belongs | Can only view readers in the groups to which the user belongs. |

To manage the RFID Management Console users, select the appropriate icon. The icons are defined in the following table.

**TABLE 5-2** Action Icons for Creating RFID Management Console Users

| Icon | Description |
|---|---|
| ⊕ | Add groups to an existing user. |
| ⊖ | Remove groups from a user. |
| 🔍 | View the list of groups to which a user belongs. |
| ✎ | Edit the user information, including the password. |
| ⊘ | Delete a user. |

## ▼ To Create New RFID Management Console Users

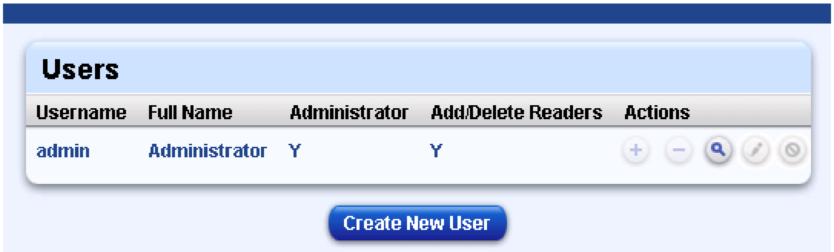
Only administrators can create new users.

1. **Log in to the RFID Management Console as the** `admin` **user.**

   See "To Access the RFID Management Console" on page 82.

2. **Choose Administration → Users.**

   The existing default `admin` user information appears.

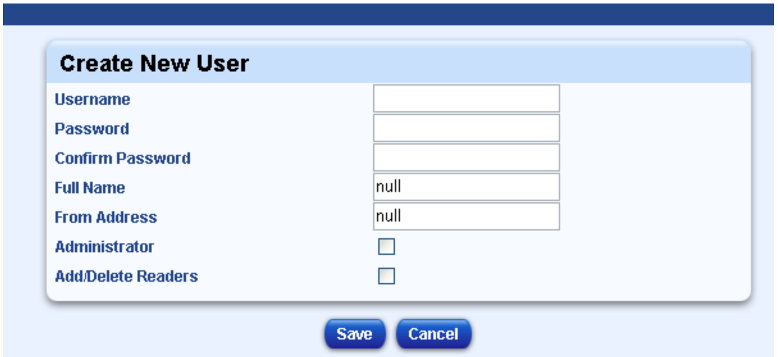


**Users**

| Username | Full Name | Administrator | Add/Delete Readers | Actions |
|---|---|---|---|---|
| admin | Administrator | Y | Y | + − 🔍 ✎ ⊘ |

3. **Click Create New User.**

   The Create New User dialog box appears.



**Create New User**

| | |
|---|---|
| Username | |
| Password | |
| Confirm Password | |
| Full Name | null |
| From Address | null |
| Administrator | □ |
| Add/Delete Readers | □ |

4. **Type the necessary values to create the new user.**

5. **Select the Administrator check box to enable the user to create new users and groups.**

6. **Select the Add/Delete Readers check box to enable the user to add and delete readers.**

# Creating and Managing Reader Groups

The Groups menu option enables you to group different readers for easier tracking. Choosing this option displays the groups that have been created and the number of readers that are in each group. A default group named All Readers contains a list of all available readers. The All Readers group enables administrators to view all the available readers in the entire system. A nonadministrator can only view groups to which he or she has been assigned

To manage the RFID Management Console groups, select the appropriate icon. The icons are defined in the following table.

**TABLE 5-3** Action Icons for Creating RFID Management Console Groups

| Icon | Description |
|---|---|
| ⊕ | Add readers to this group. |
| ⊖ | Delete readers from this group. |
| 🔍 | View the list of readers assigned to this group. |
| ✎ | Edit the group name and description. |
| ⊘ | Delete this group. |

## ▼ To Create a RFID Management Console Reader Group

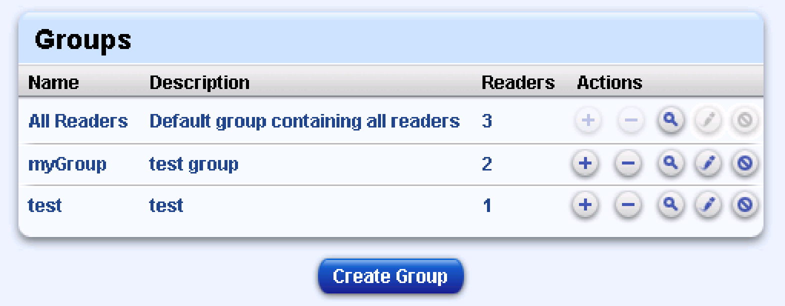
Only administrators can create groups.

1. **Log in to the RFID Management Console as the `admin` user.**

   See "To Access the RFID Management Console" on page 82.

2. **Choose Administration → Groups.**

   The following screen appears and shows the default group, All Readers, and any previously created groups and the number of readers that are in each group.



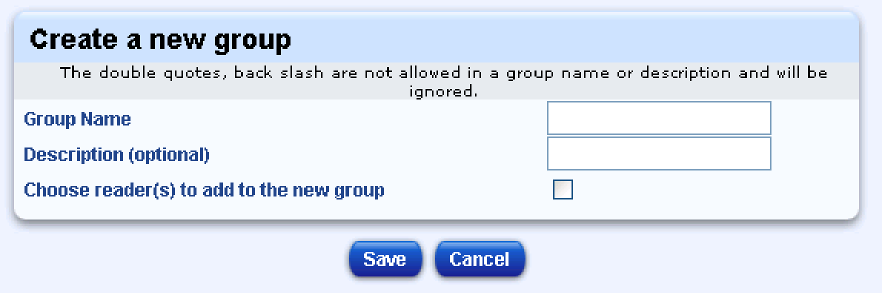
Groups

| Name | Description | Readers | Actions |
|---|---|---|---|
| All Readers | Default group containing all readers | 3 | + − 🔍 ✎ ⊘ |
| myGroup | test group | 2 | + − 🔍 ✎ ⊘ |
| test | test | 1 | + − 🔍 ✎ ⊘ |

If no groups have been created, only the default All Readers group appears. If the user is not assigned to any groups, then no groups appear.

3. **Click Create Group.**

   The following screen appears:



Create a new group

The double quotes, back slash are not allowed in a group name or description and will be ignored.

| | |
|---|---|
| Group Name | |
| Description (optional) | |
| Choose reader(s) to add to the new group | □ |

4. **Type the Group Name**

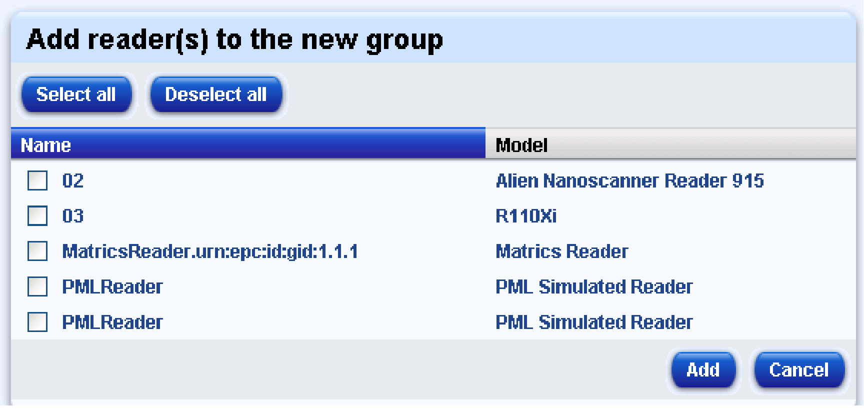
5. **(Optional) Type the Description.**

6. **(Optional) Select the check box, Choose reader(s) to add to the new group.**

   Selecting this check box enables you to add readers to the new group at the same time as you create the group. If the check box is not selected, only the group is created and the readers must be added later.

7. **Click Save.**

   - If the reader check box is not selected, the Create Group screen appears again. You can add your readers at a later time. You have completed this procedure.
   - If the reader check box is selected, the following screen appears. Proceed to Step 8.



| | Name | Model |
|---|---|---|
| □ | 02 | Alien Nanoscanner Reader 915 |
| □ | 03 | R110Xi |
| □ | MatricsReader.urn:epc:id:gid:1.1.1 | Matrics Reader |
| □ | PMLReader | PML Simulated Reader |
| □ | PMLReader | PML Simulated Reader |

8. **Select the readers to add to the group, then click Add.**

   Your new group appears in the list of groups.

9. **After you have created your groups, you can log out of the RFID Management Console by choosing Logout from the menu.**

## Managing RFID Management Console Settings

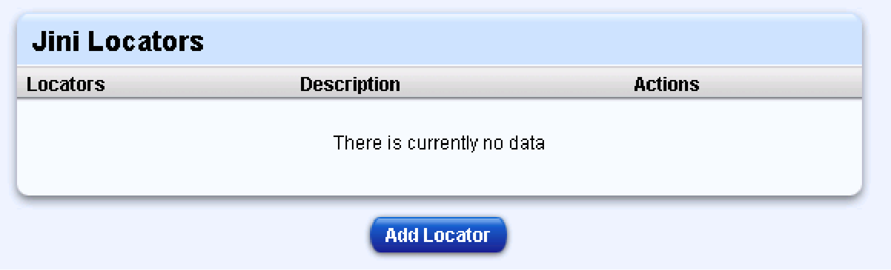
The Settings menu shows the following options:

- **Jini Locators** – You can add Jini locators to the Jini lookup server. By default, a reader client can only find those readers running on RFID Execution Agents and Control Stations that are within the same subnet. You can extend the search for readers outside this subnet by adding locators. These locators are saved across system restarts so that each time you start up the system, the Jini lookup server starts with the saved configuration. You can also delete previously created Jini locators. See the *Sun Java System RFID Software 3.0 Developer's Guide* for more information on using the RFID reader client API. For a detailed discussion of how Jini locators are used, see the following tutorial: `http://www.cswl.com/whiteppr/tutorials/jini.html`. Look at the following sections in the tutorial:
  - Discovering a Lookup Service
  - Unicast Discovery
- **Email Notification** – You can receive email notification for alerts from the RFID reader network. The alerts tell you if a reader or other RFID device is operational or not.

## ▼ To Add a Jini Locator

1. **Log in to the RFID Management Console.**
2. **Choose Administration → Settings.**

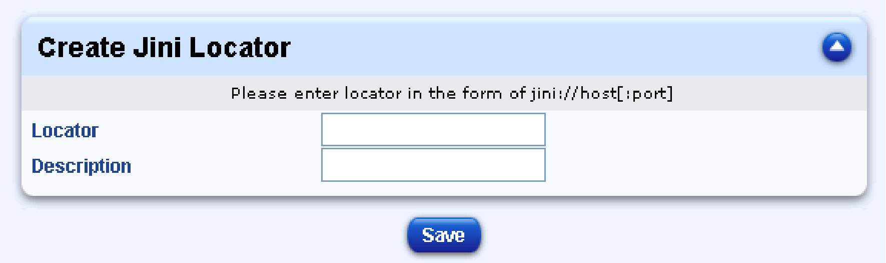
   The Jini Locators list appears.



| Locators | Description | Actions |
|---|---|---|

There is currently no data

3. **Click Add Locator.**



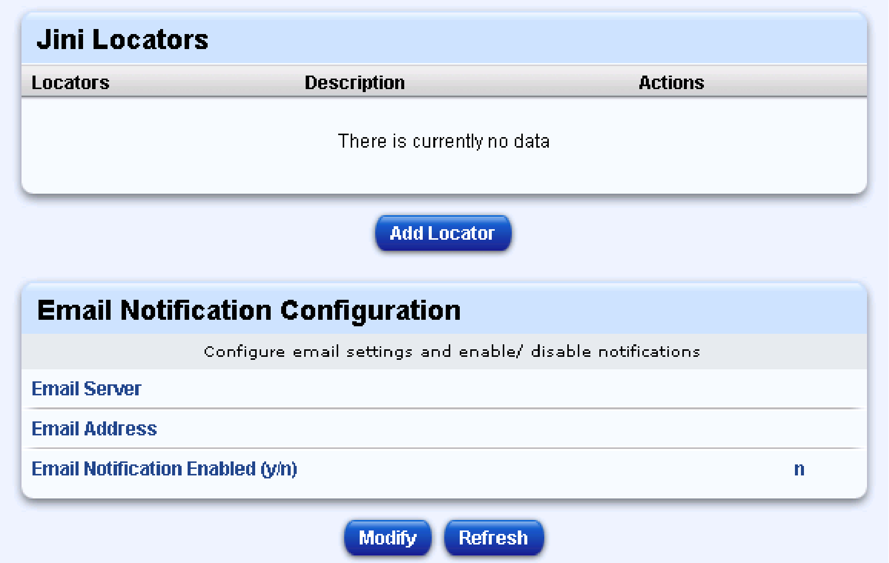
Please enter locator in the form of jini://host[:port]

4. **Type the locator in the format** `jini://`*IP-address*:*port-number.*
5. **(Optional) Type a description for the locator.**
6. **Click Save.**

## ▼ To Receive Email Notifications for Alerts

1. **Log in to the RFID Management Console.**
2. **Choose Administration → Settings.**

   The Email Notification Configuration dialog box appears. This dialog box appears with the Jini Locator dialog box described in "To Add a Jini Locator" on page 90.



Jini Locators

| Locators | Description | Actions |
|---|---|---|
| There is currently no data | | |

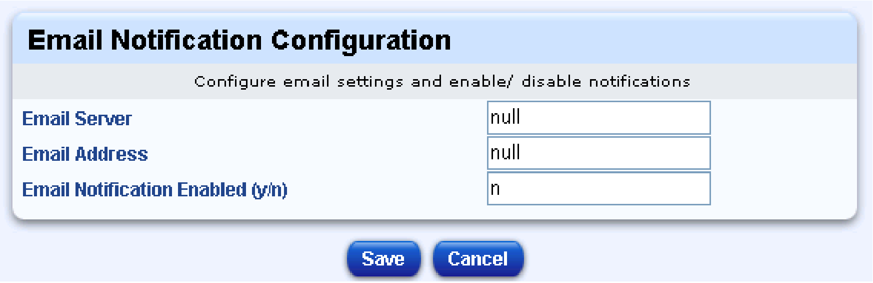
Email Notification Configuration

Configure email settings and enable/ disable notifications

| | |
|---|---|
| Email Server | |
| Email Address | |
| Email Notification Enabled (y/n) | n |

3. **Click Modify.**

   The Email Notification Configuration dialog box appears.



Email Notification Configuration

Configure email settings and enable/ disable notifications

| | |
|---|---|
| Email Server | null |
| Email Address | null |
| Email Notification Enabled (y/n) | n |

4. **Type your email server.**

   For example, `mail-server.company.com`.

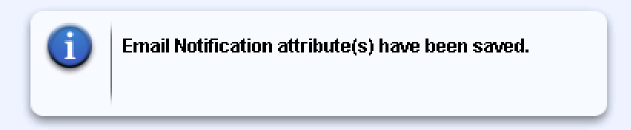
5. **Type your email address.**

   For example, `myname@company.com`.

6. **Enable the Alerts email notification by typing `y` in the Email Notification Enabled (y/n) text box.**

7. **Click Save.**

   You are returned to the original Settings page and receive the following confirmation message.



Email Notification attribute(s) have been saved.

## Changing the `admin` User Password

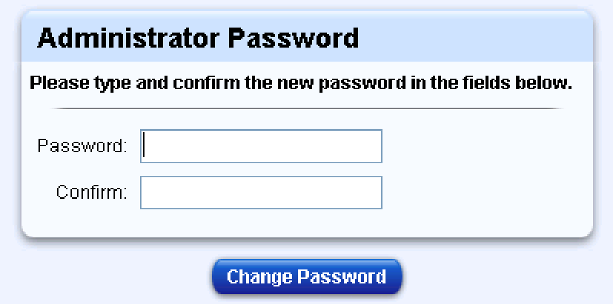
Use the following procedure to change the default password that is used to access the RFID Management Console as the `admin` user.

### ▼ To Change the `admin` User Password

1. **Log in to the RFID Management Console as the `admin` user.**

2. **Choose Administration → Password.**

   The following dialog box appears.



Administrator Password

Please type and confirm the new password in the fields below.

3. **Type your new password.**

4. **Type the same value in the Confirm text field.**

5. **Click Change Password to save the new value.**

   Use the new password the next time that you log in to the RFID Management Console as the `admin` user.

# Performing RFID Reader Network Management Tasks

The RFID Management Console provides the following options for managing the RFID reader network. Expand the RFID menu option to see the following links:

- RFID Readers
- Components (only available to administrative users)

## RFID Reader Network Grouping Function

The RFID Management Console provides a reader grouping function so that distinct groups of readers can be monitored in a meaningful way. For example, all readers in a particular section of a warehouse might be grouped together and specified with a the name `warehouseSectionA`. Another area of the warehouse might contain a set of readers in a group named `warehouseShippingArea`. Once you have grouped your readers and other RFID devices into meaningful groups, the viewing functions provided by the RFID Management Console can be used effectively to monitor the RFID reader network.

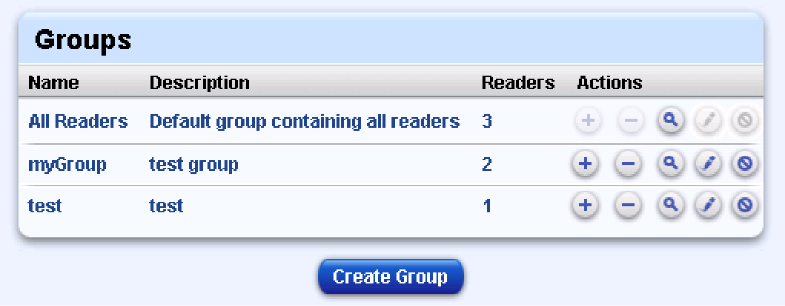
Due to the dynamic nature of RFID readers, a short-lived network malfunction can cause a reader to not be seen for a period of time. Deleting a malfunctioning reader from its group would require it to be added back to the group manually when it was reactivated. Rather than delete such a reader from its group, every reader that has ever been added to a group is shown. If, for some reason, a reader disappears from the network, it is marked as inactive. An inactive reader is indicated by greying out (or disabling) the Status icon with the horizontal dash. This icon is shown in TABLE 5-5.

You can view the reader groups in two ways as described in the following procedures.

## ▼ To View Readers Using the Administration View

1. **From the RFID Management Console main menu choose Administration → Groups.**

   The Groups page appears.

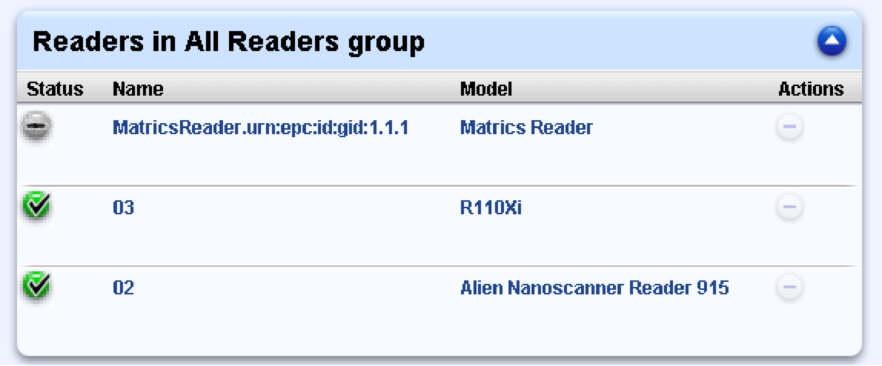


## Groups

| Name | Description | Readers | Actions |
|---|---|---|---|
| All Readers | Default group containing all readers | 3 | + − 🔍 ✎ ⊘ |
| myGroup | test group | 2 | + − 🔍 ✎ ⊘ |
| test | test | 1 | + − 🔍 ✎ ⊘ |

2. **Choose a reader group to view.**

   You see something similar to the following screen capture. In this example one reader is inactive and two readers are active. The delete Action icon is greyed out.



## Readers in All Readers group

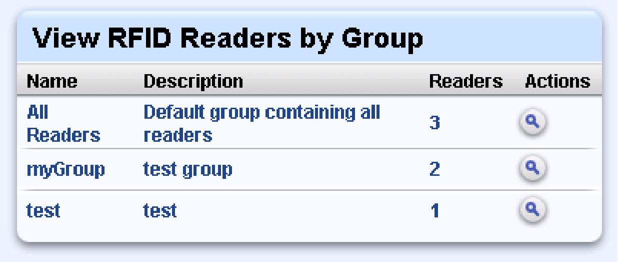
| Status | Name | Model | Actions |
|---|---|---|---|
| − | MatricsReader.urn:epc:id:gid:1.1.1 | Matrics Reader | − |
| ✓ | 03 | R110Xi | − |
| ✓ | 02 | Alien Nanoscanner Reader 915 | − |

## ▼ To View Readers Using the RFID Readers View

The RFID Readers view enables you to view readers by group and to view and to set the reader attributes. You can quickly view the status of all devices.

1. **From the RFID Management Console main menu choose RFID → RFID Readers.**

   The View Readers by Group page appears as shown in the following screen capture. The page shows the existing reader groups to which you belong. If no groups are assigned to you (your user login) no groups appear on this screen.

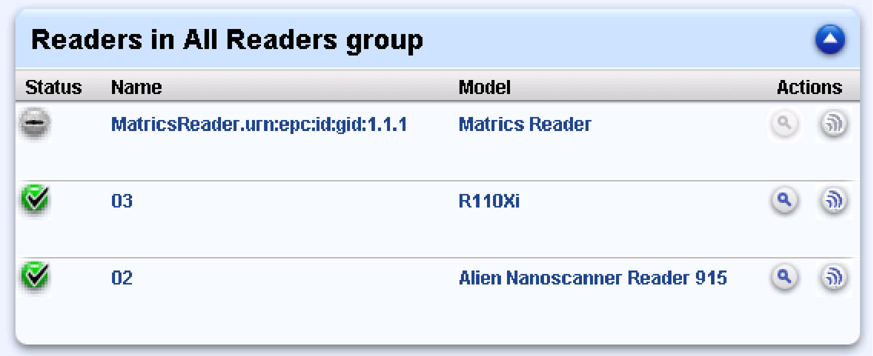


View RFID Readers by Group

| Name | Description | Readers | Actions |
|---|---|---|---|
| All Readers | Default group containing all readers | 3 | 🔍 |
| myGroup | test group | 2 | 🔍 |
| test | test | 1 | 🔍 |

2. **To view the readers in a group, click the Inspect icon in the Actions column.**

   For example, if you click the Inspect icon for the All Readers group, the Readers in All Readers group panel appears as shown in the following screen capture. The group has one inactive reader and two active readers. The Actions icons for the inactive readers are greyed out because you cannot inspect them or view their tags when they are inactive. If the readers become active again later, then they would appear with the green Status icon and be selectable for other actions. TABLE 5-4 describes the Actions icons.



Readers in All Readers group

| Status | Name | Model | Actions |
|---|---|---|---|
| ⊖ | MatricsReader.urn:epc:id:gid:1.1.1 | Matrics Reader | 🔍 📶 |
| ✓ | 03 | R110Xi | 🔍 📶 |
| ✓ | 02 | Alien Nanoscanner Reader 915 | 🔍 📶 |

**TABLE 5-4** Actions Icons for RFID Readers

| Icon | Description |
|---|---|
| 🔍 | Inspect – View the attributes for this reader. |
| 📶 | View Tags – Display the RFID tags that are within view of the selected reader. |

3. **Review the status of the readers in the selected group.**

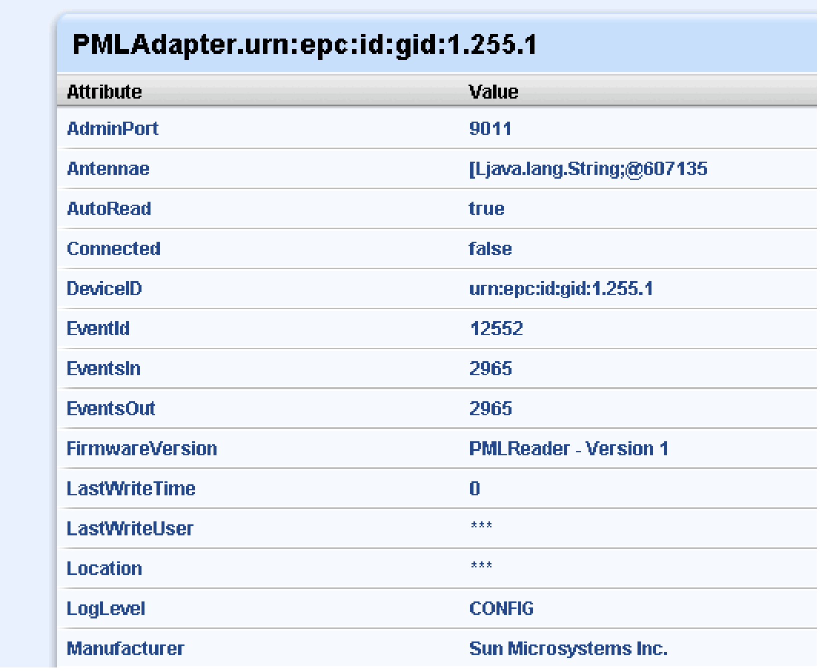
   The Status icons are described in the following table.

**TABLE 5-5** Status Indicator Icons for RFID Readers

| Icon | Description |
|---|---|
| ✓ | The reader is connected. |
| ! | The reader is disconnected. |
| ••• | The reader is in the “other” state. |
| ✗ | The reader is not responding. |
| − | The reader is inactive. |

4. **To view reader attributes, click the Inspect icon (see** TABLE 5-4**) in the Actions column.**

   The RFID Reader Attribute pane appears.



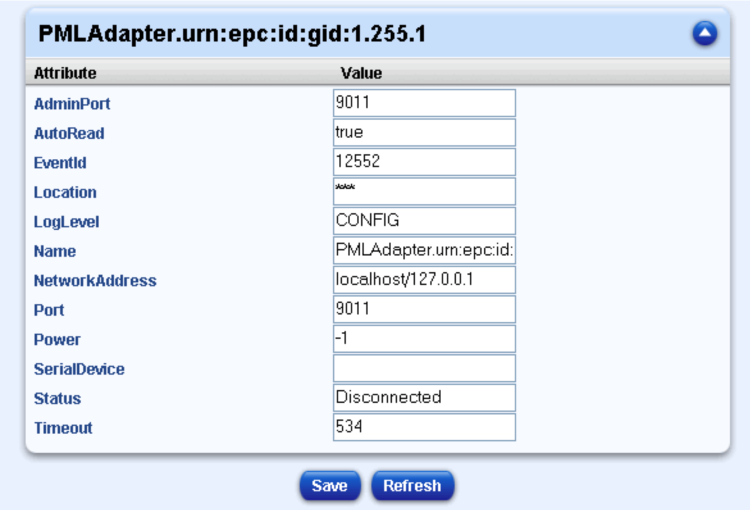
PMLAdapter.urn:epc:id:gid:1.255.1

| Attribute | Value |
|---|---|
| AdminPort | 9011 |
| Antennae | [Ljava.lang.String;@607135 |
| AutoRead | true |
| Connected | false |
| DeviceID | urn:epc:id:gid:1.255.1 |
| EventId | 12552 |
| EventsIn | 2965 |
| EventsOut | 2965 |
| FirmwareVersion | PMLReader - Version 1 |
| LastWriteTime | 0 |
| LastWriteUser | *** |
| Location | *** |
| LogLevel | CONFIG |
| Manufacturer | Sun Microsystems Inc. |

5. (Optional) To modify reader properties, follow these steps:

   a. Scroll to the bottom of the Reader Attribute pane, and click Modify.

      A pane that shows the modifiable properties appears.



PMLAdapter.urn:epc:id:gid:1.255.1

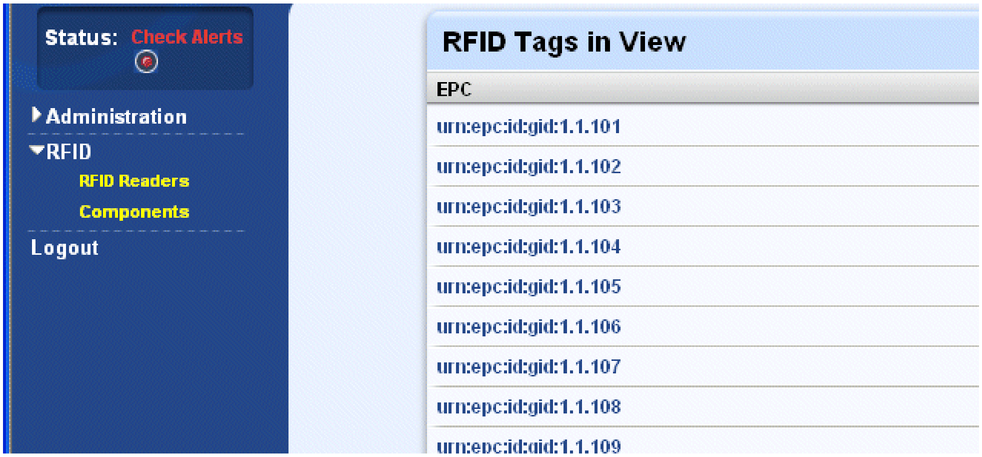
| Attribute | Value |
|---|---|
| AdminPort | 9011 |
| AutoRead | true |
| EventId | 12552 |
| Location | *** |
| LogLevel | CONFIG |
| Name | PMLAdapter.urn:epc:id: |
| NetworkAddress | localhost/127.0.0.1 |
| Port | 9011 |
| Power | -1 |
| SerialDevice | |
| Status | Disconnected |
| Timeout | 534 |

   b. Type your changes, and click Save.

c. **(Optional) Click Refresh to determine if attribute values have changed while you have been reviewing them.**

6. **(Optional) To view the tags in view of a selected reader, from the Readers in Groups pane (shown in Step 2), click the View Tags icon located in the Actions column.**

   The RFID Tags in View pane appears.



RFID Tags in View

| EPC |
|---|
| urn:epc:id:gid:1.1.101 |
| urn:epc:id:gid:1.1.102 |
| urn:epc:id:gid:1.1.103 |
| urn:epc:id:gid:1.1.104 |
| urn:epc:id:gid:1.1.105 |
| urn:epc:id:gid:1.1.106 |
| urn:epc:id:gid:1.1.107 |
| urn:epc:id:gid:1.1.108 |
| urn:epc:id:gid:1.1.109 |

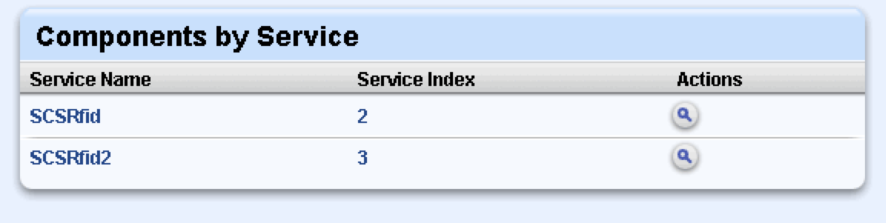
## ▼ To View RFID Components

The components menu option enables you to view all RFID system components, such as filters, connectors, and reader adapters, for each running service. This option is only available to the administrative user.

1. **Log in to the RFID Management Console as the** `admin` **user.**

2. **Choose RFID → Components.**

   A list of RFID services similar to the following appears.

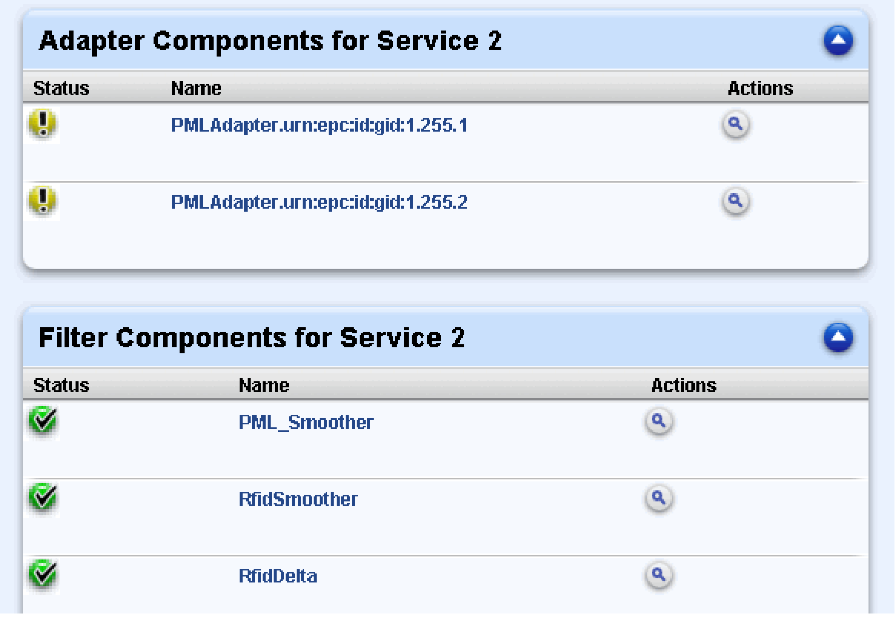


## Components by Service

| Service Name | Service Index | Actions |
|---|---|---|
| SCSRfid | 2 | 🔍 |
| SCSRfid2 | 3 | 🔍 |

3. **To view the components for a specific service, click the Inspect icon in the Actions column.**

   Each component type is grouped in its own table, similar to the following.



## Adapter Components for Service 2

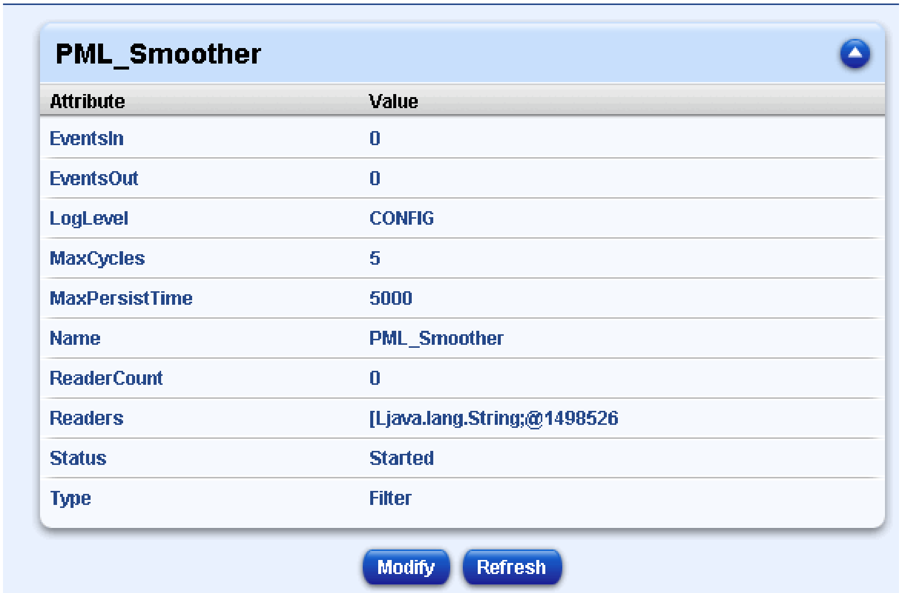
| Status | Name | Actions |
|---|---|---|
| ! | PMLAdapter.urn:epc:id:gid:1.255.1 | 🔍 |
| ! | PMLAdapter.urn:epc:id:gid:1.255.2 | 🔍 |

## Filter Components for Service 2

| Status | Name | Actions |
|---|---|---|
| ✓ | PML_Smoother | 🔍 |
| ✓ | RfidSmoother | 🔍 |
| ✓ | RfidDelta | 🔍 |

4. **(Optional) To view the component properties, click the Inspect icon in the Actions column.**

   For example, the following screen capture shows the properties for the `PML_Smoother` component.



PML_Smoother

| Attribute | Value |
|---|---|
| EventsIn | 0 |
| EventsOut | 0 |
| LogLevel | CONFIG |
| MaxCycles | 5 |
| MaxPersistTime | 5000 |
| Name | PML_Smoother |
| ReaderCount | 0 |
| Readers | [Ljava.lang.String;@1498526 |
| Status | Started |
| Type | Filter |

# Monitoring the Status of Your RFID Reader Network By Using Alerts

The RFID Management Console enables you to view alerts from the RFID reader network. An alert indicates that one of the RFID readers or other RFID devices is malfunctioning. The red Check Alerts label and the button below it indicate that there are alerts to be viewed. When you review and dismiss the alert from the Alerts display, the alert is archived in a database. When all alerts have been archived, the Check Alerts button turns green to indicate no more alerts exist. When new alerts arrive, the Check Alerts button turns red again.

You can also create alert filters. The RFID Software 3.0 supports two types of alert filters:

- **Date Range** – Displays all alerts within the specified period of time
- **Elapsed Time** – Displays all alerts in the past *x* number of minutes

## ▼ To Create and Use Alert Filters

1. **From the RFID Management Console, choose Settings → Alert Filters.**

   A list of the existing filters appears. By default, there is only one filter, the "Last 24 hours" filter. The default state for this filter is OFF, which is indicated by the Status icon, indicated by a red circle with an X.

2. **Click Create New Filter.**

3. **Check the type of filter that you want to create.**

   - To create a Date Range filter, follow these steps:

   a. **Type a name for the filter, and click Next.**

   b. **Select Date Range and type the following properties values:**

      - `startDate`
      - `description`
      - `endDate`

      Type two different dates with the end date later than the start date. If you type the same date, you will not view any alerts.

   c. **Click Save.**

      The new filter appears in the Alert Filters list.

---

**Note –** Only one filter is active at a time.

---

   - To create an Elapsed Time filter, follow these steps:

   a. **Type a name for the filter, and click Next.**

   a. **Select Elapsed Time and type the following properties values:**

      - `elapsedTime` – an integer that represents number of minutes
      - `description` – a unique name to described this filter

   b. **Click Save.**

      The new filter appears in the Alert Filters list.

4. **To activate a filter, click the green check icon in the Actions column.**

   Activating a filter deactivates any previously active filter and activates the filter you selected. Once you click the Activate icon, the Status column changes from a red X to a green check icon.

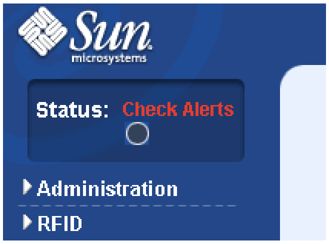
5. **To view the list of alerts as filtered by the active filter, click Check Alerts in the Status area of the left menu.**

   The RFID Management Console uses the currently active filter to display the alerts.

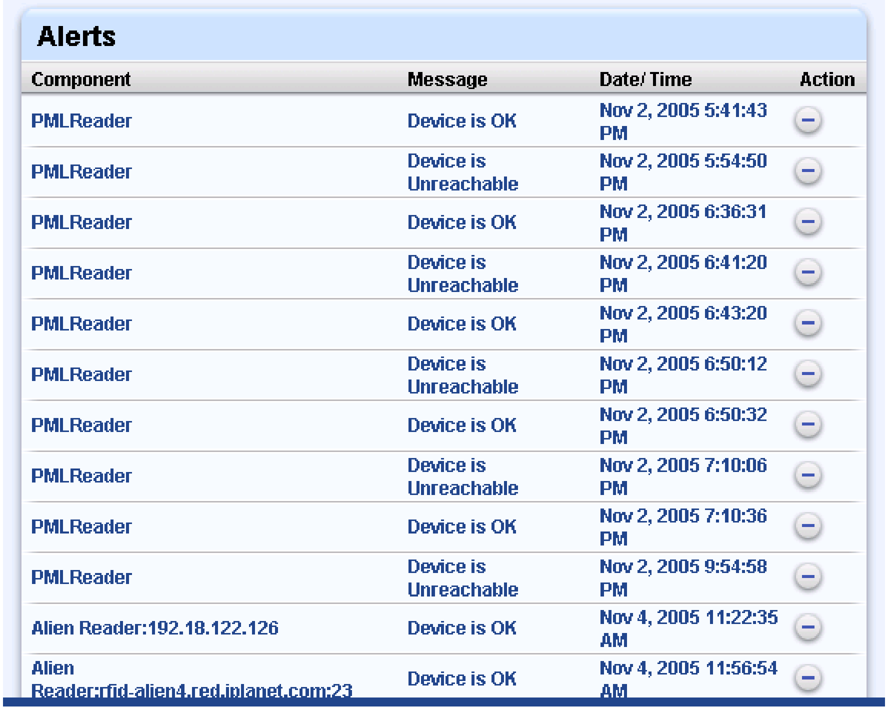
6. **To archive these alerts, click Archive All.**

## ▼ To View Alerts

- Click the Check Alerts button found in the upper left-hand corner of the Management Console.



The Alerts pane appears showing the alerts for the currently active filter. You see something similar to the following screen capture.



Alerts

| Component | Message | Date/ Time | Action |
|---|---|---|---|
| PMLReader | Device is OK | Nov 2, 2005 5:41:43 PM | − |
| PMLReader | Device is Unreachable | Nov 2, 2005 5:54:50 PM | − |
| PMLReader | Device is OK | Nov 2, 2005 6:36:31 PM | − |
| PMLReader | Device is Unreachable | Nov 2, 2005 6:41:20 PM | − |
| PMLReader | Device is OK | Nov 2, 2005 6:43:20 PM | − |
| PMLReader | Device is Unreachable | Nov 2, 2005 6:50:12 PM | − |
| PMLReader | Device is OK | Nov 2, 2005 6:50:32 PM | − |
| PMLReader | Device is Unreachable | Nov 2, 2005 7:10:06 PM | − |
| PMLReader | Device is OK | Nov 2, 2005 7:10:36 PM | − |
| PMLReader | Device is Unreachable | Nov 2, 2005 9:54:58 PM | − |
| Alien Reader:192.18.122.126 | Device is OK | Nov 4, 2005 11:22:35 AM | − |
| Alien Reader:rfid-alien4.red.iplanet.com:23 | Device is OK | Nov 4, 2005 11:56:54 AM | − |

## ▼ To Modify Alert Filters

1. **Click the Inspect (magnifying glass) icon next to a filter.**

   The Edit Filter pane appears.

2. **Type the new property values, and click Save.**

## ▼ To Delete Alert Filters

- **Click the Delete icon (a circle with a minus sign).**

  The filter is immediately deleted. No undo function is available.

## ▼ To Turn On Alert Filters

- **Click the green check icon or the red X icon in the Status column to toggle the filter's state to on or off.**

CHAPTER 6

# Sending Events From the RFID Event Manager to the RFID Information Server

The easiest way to send the RFID tag events from the RFID Event Manager to a database is to use the Epcis connector (a component of the RFID Event Manager), the RFID Information Server (also known as EPCIS), and an application server. There are other ways to accomplish this task, but they require more programming.

The Epcis connector can be used with two protocols: the Java Message Service (JMS) API and HTTP. The resulting connectors are called the EpcisHttp connector and the EpcisJms connector.

This chapter includes the following topics:

- Prerequisites for Using the Epcis Connectors
- Enabling Usage of JMS With the RFID Information Server
- Configuring the RFID Event Manager to Use an EpcisJms Connector
- Configuring the RFID Event Manager to Use an EpcisHttp Connector

## Prerequisites for Using the Epcis Connectors

The following software must be installed and properly configured before you can use the procedures in this chapter.

- RFID Event Manager – See *Sun Java System RFID Software 3.0 Installation Guide*.
- A supported database – See the documentation for your database
- A supported application server – See the documentation for your application server.
- JDBC Drivers – See the *Sun Java System RFID Software 3.0 Installation Guide*.
- RFID Information Server – See the *Sun Java System RFID Software 3.0 Installation Guide*.

- (Required to use JMS) Configuring the JMS environment – If you are using JMS queues or topics as the message transport protocol between the RFID Event Manager and the RFID Information Server, you must set up the JMS environment. See "Enabling Usage of JMS With the RFID Information Server" on page 106.

# Enabling Usage of JMS With the RFID Information Server

Sun Java System Application Server 8.1 (Application Server) provides support for applications that use the Java Message Service (JMS) application programming interface (API) for messaging operations. JMS is a set of programming interfaces that provide a common way for Java applications to create, send, receive, and read messages in a distributed environment. JMS is a standard way that Java 2 Platform, Enterprise Edition (J2EE™ platform) applications perform asynchronous messaging. J2EE components, web components or Enterprise JavaBeans™ (EJB™) components - can use the JMS API to send messages that can be consumed asynchronously by a specialized EJB, called a Message Driven Bean (MDB).

An MDB is similar to a session bean, except it responds to a JMS message rather than an RMI event. MDBs were introduced in the EJB 2.0 specification. The MDB represents the integration of JMS (Java Message Service) with the EJB software to create an entirely new type of bean designed to handle asynchronous JMS messages.

Application Server support for JMS messaging and for MDBs, requires messaging middleware, a JMS provider, that implements the JMS specification. Application Server uses Sun Java™ System Message Queue 3 2005Q1 (Message Queue) as its native JMS provider. The Sun Java System RFID Information Server installer creates the following JMS resources on the application server:

**TABLE 6-1** Java Message Service Resources

| JMS Resource | JNDI Name |
|---|---|
| Topic Connection Factory | `jms/TopicConnectionFactory` |
| Topic | `jms/epcisTopic` |
| Physical Destination | `epcisTopic` |
| Queue Connection Factory | `jms/QueueConnectionFactory` |
| Queue | `jms/epcisQueue` |
| Physical Destination | `epcisQueue` |

RFID Information Server clients can be configured to use the JMS interface using a file system JNDI provider (Java Naming and Directory Interface).

This section include the following topics:

- Sun Java System Message Queue Documentation
- Enabling Usage of JMS with Sun Java System Application Server 8.1

## Sun Java System Message Queue Documentation

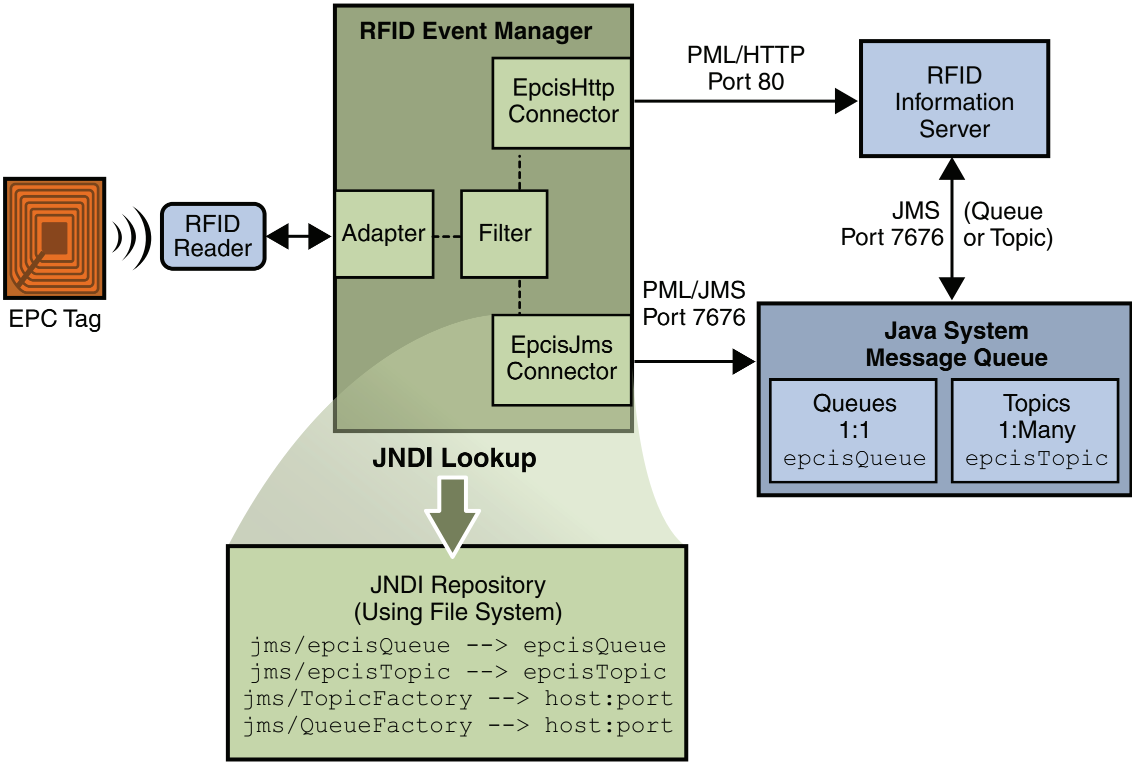
The *Sun Java System Message Queue 3 2005Q4 Administration Guide* can be found at `http://docs.sun.com/app/docs/doc/819-2571`.

The complete documentation set for Sun Java System Message Queue 3 2005Q4 can be found at `http://docs.sun.com/app/docs/coll/1307.1`.

## Enabling Usage of JMS with Sun Java System Application Server 8.1

The RFID Information Server installer creates the necessary JMS resources on the Application Server. You need to configure the Java Naming and Directory Interface™ (JNDI) File System Provider object store.

See *Sun Java System Application Server Platform Edition 8.1 2005Q2 Update 2 Administration Guide* found at `http://docs.sun.com/app/docs/doc/819-2641` for more information on configuring JMS Resources and understanding JNDI resources. It is not the intent of this guide to fully explain these topics. The following illustration shows how the RFID Event Manager queries the JNDI repository to determine the physical parameters necessary for using the JMS messaging protocol.



The procedures in this chapter show screen captures using Application Server Platform Edition on a Windows system. Using the Application Server and Message Queue GUI interfaces on other platforms may differ slightly in appearance.

Perform the following procedures in the order they are presented:

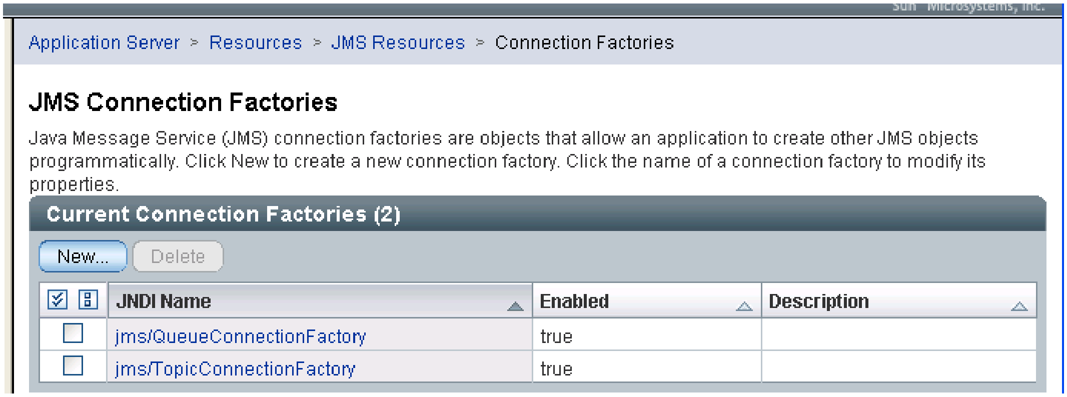
- "To Confirm the JMS Resources on the Application Server" on page 108
- "To Add the Broker Using Sun Java System Message Queue" on page 109
- "To Add an Object Store for RFID Information Server" on page 112
- "To Add a Topic Connection Factory" on page 114
- "To Add a Queue Connection Factory" on page 116
- "To Add the EPCIS Destination Objects" on page 117

## ▼ To Confirm the JMS Resources on the Application Server

The RFID Information Server installer creates JMS resources for you. This procedure describes how to view and confirm that these resources were successfully created.

1. **Start the application server default server instance.**
2. **Log in to the application server admin console.**
3. **Choose Resources → JMS Resources → Connection Factories.**

   Confirm that the `jms/QueueConnectionFactory` and `jsm/TopicConnectionFactory` nodes are present as shown in the following screen capture.



Application Server > Resources > JMS Resources > Connection Factories

JMS Connection Factories

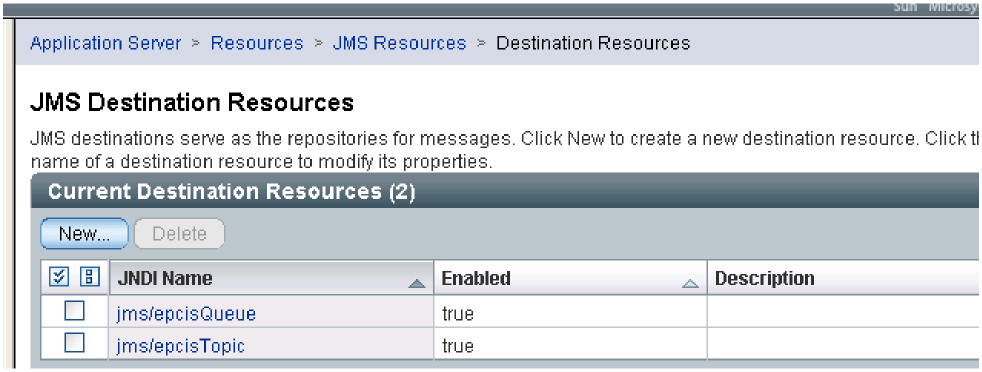
Java Message Service (JMS) connection factories are objects that allow an application to create other JMS objects programmatically. Click New to create a new connection factory. Click the name of a connection factory to modify its properties.

Current Connection Factories (2)

| ☑ | JNDI Name | Enabled | Description |
|---|---|---|---|
| ☐ | jms/QueueConnectionFactory | true | |
| ☐ | jms/TopicConnectionFactory | true | |

4. **Choose Resources → JMS Resources → Destination Resources nodes.**

   Confirm that the `jms/epcisTopic` and `jsm/epcisQueue` nodes are present as shown in the following screen capture.



Application Server > Resources > JMS Resources > Destination Resources

JMS Destination Resources

JMS destinations serve as the repositories for messages. Click New to create a new destination resource. Click tl name of a destination resource to modify its properties.

Current Destination Resources (2)

| ☑ | JNDI Name | Enabled | Description |
|---|---|---|---|
| ☐ | jms/epcisQueue | true | |
| ☐ | jms/epcisTopic | true | |

## ▼ To Add the Broker Using Sun Java System Message Queue

You need to a add a broker that points to your local JMS Message Queue server, or whichever Message Queue server you want to use.

---

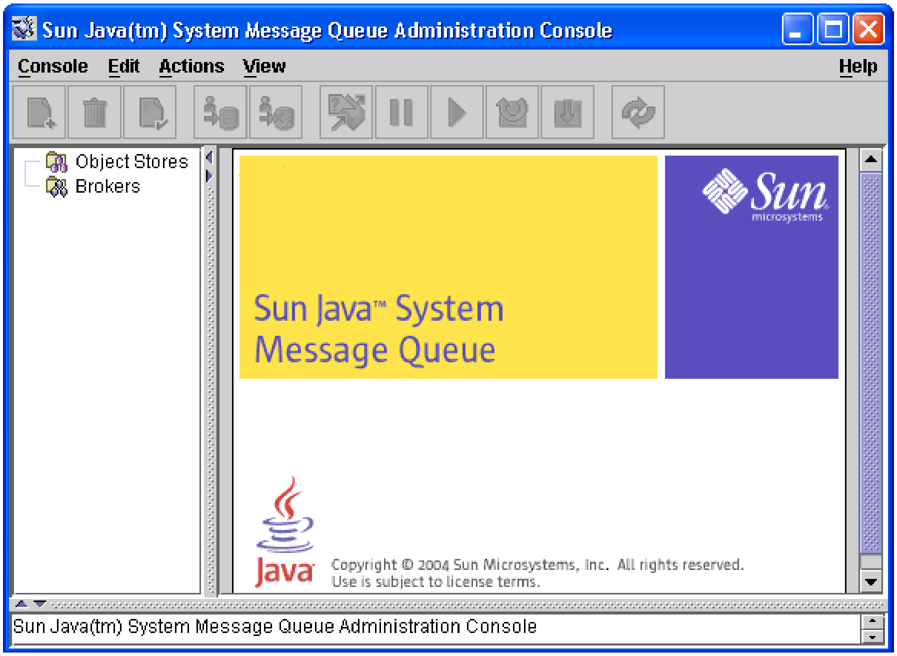
**Note –** This example uses the Sun Java System Message Queue that is installed as part of the Sun Java System Application Server 8.1 installation.

---

1. **Start the Message Queue admin console.**

   These examples use the default Application Server installation directory for each platform.

   - Solaris OS (Application Server Enterprise Edition) – `/opt/SUNWappserver/imq/bin/imqadmin`
   - Solaris OS (Application Server Platform Edition) – `/opt/SUNWappserver/imq/bin/imqadmin`
   - Red Hat Linux (Application Server Enterprise Edition) – `/opt/sun/mq/bin/imqadmin`
   - Red Hat Linux (Application Server Platform Edition) – `/opt/SUNWappserver/imq/bin/imqadmin`
   - Windows (Application Server, both Platform Edition and Enterprise Edition) – `C:\Sun\AppServer\imq\bin\imqadmin.exe`



Sun Java(tm) System Message Queue Administration Console

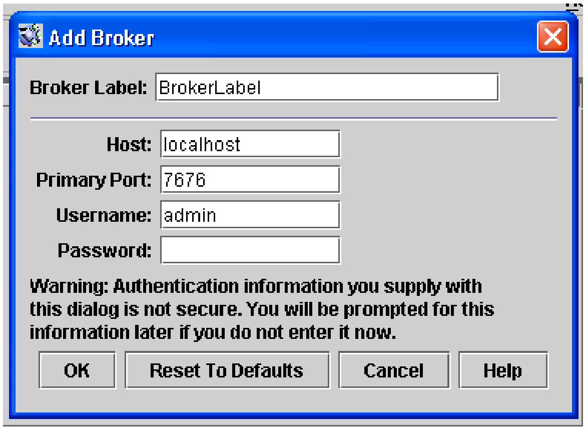
Copyright © 2004 Sun Microsystems, Inc. All rights reserved.
Use is subject to license terms.

Sun Java(tm) System Message Queue Administration Console

2. **Select the Brokers node in the navigation tree.**

3. **From the menu, choose Actions → Add Broker.**

   The Add Broker dialog box appears.



Broker Label: BrokerLabel

Host: localhost

Primary Port: 7676

Username: admin

Password:

Warning: Authentication information you supply with this dialog is not secure. You will be prompted for this information later if you do not enter it now.

4. **Add a new broker by following these steps:**

   a. **In the Broker Label field, type a name for your broker.**

      For this example, type `TestRFIDBroker`.

   a. **In the Host field, type the IP address of the machine where your Message Queue server is installed.**

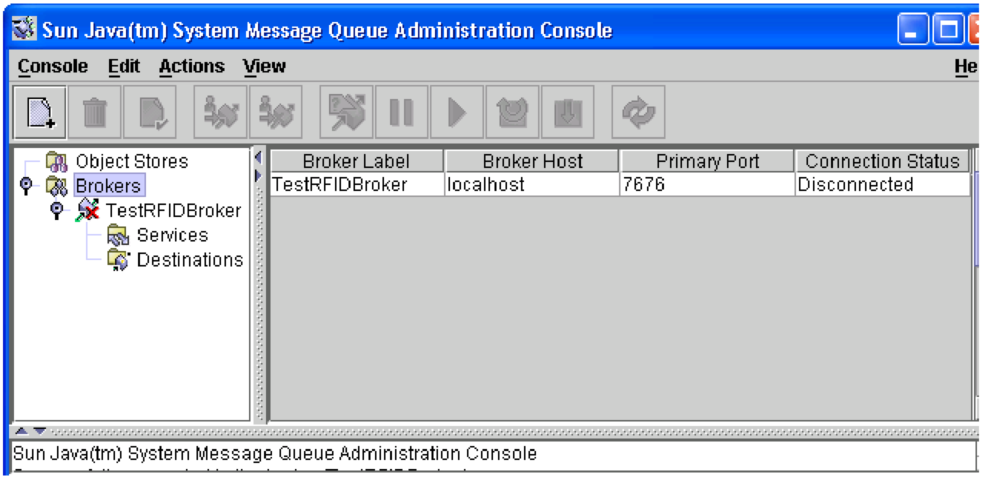
      In this example, this is the system where Application Server is installed.

   b. **In the Primary Port field, confirm the default value of 7676.**

   c. **Type the Message Queue** `admin` **username and password.**

      The `admin` username and password is `admin`.

   d. **Click OK.**

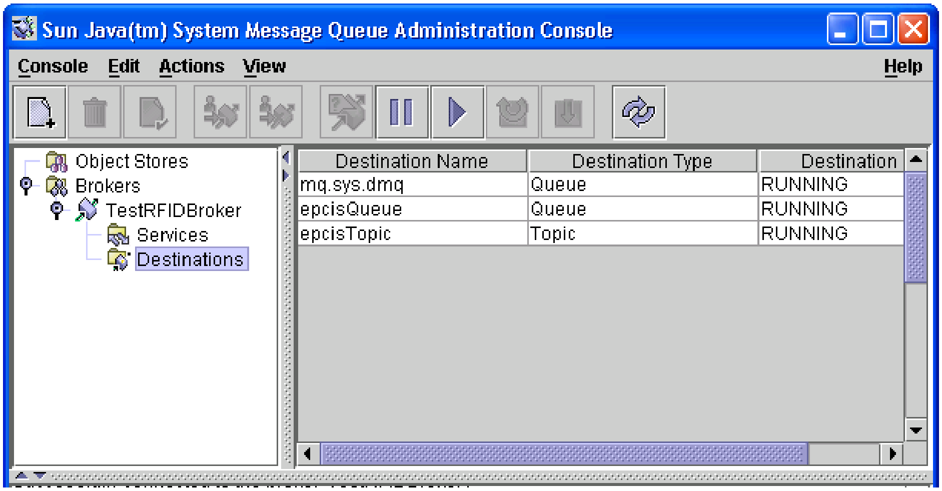


| Broker Label | Broker Host | Primary Port | Connection Status |
|---|---|---|---|
| TestRFIDBroker | localhost | 7676 | Disconnected |

Sun Java(tm) System Message Queue Administration Console

5. **Right-click the new Broker icon in the navigation tree and choose Connect to Broker.**

   For this example, right-click `TestRFIDBroker`. The destinations match the destinations that you viewed in the Application Server JMS Resources.



| Destination Name | Destination Type | Destination |
|---|---|---|
| mq.sys.dmq | Queue | RUNNING |
| epcisQueue | Queue | RUNNING |
| epcisTopic | Topic | RUNNING |

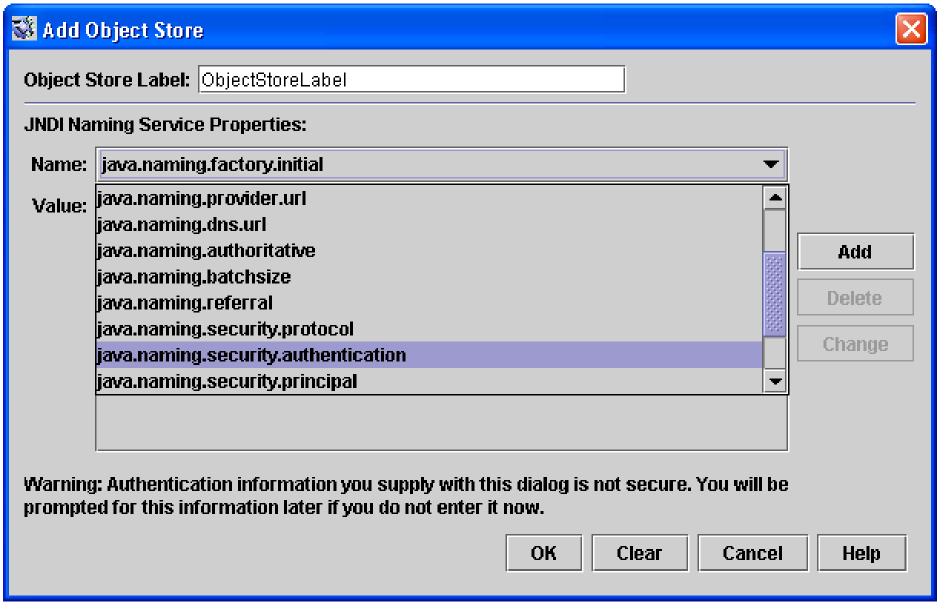
## ▼ To Add an Object Store for RFID Information Server

From the Message Queue admin console, add a new Object Store by following these steps:

1. **Select the Object Stores node in the navigation tree.**

2. **From the menu, choose Actions → Add Object Store.**

   The Add Object Store dialog box appears and enables you to add the necessary JNDI Naming Server Properties. The following screen capture shows the JNDI Naming Server Properties drop-down list.



Add Object Store

Object Store Label: ObjectStoreLabel

JNDI Naming Service Properties:

Name: java.naming.factory.initial

Value:

- java.naming.provider.url
- java.naming.dns.url
- java.naming.authoritative
- java.naming.batchsize
- java.naming.referral
- java.naming.security.protocol
- java.naming.security.authentication
- java.naming.security.principal

Warning: Authentication information you supply with this dialog is not secure. You will be prompted for this information later if you do not enter it now.

3. **Add the following fields:**

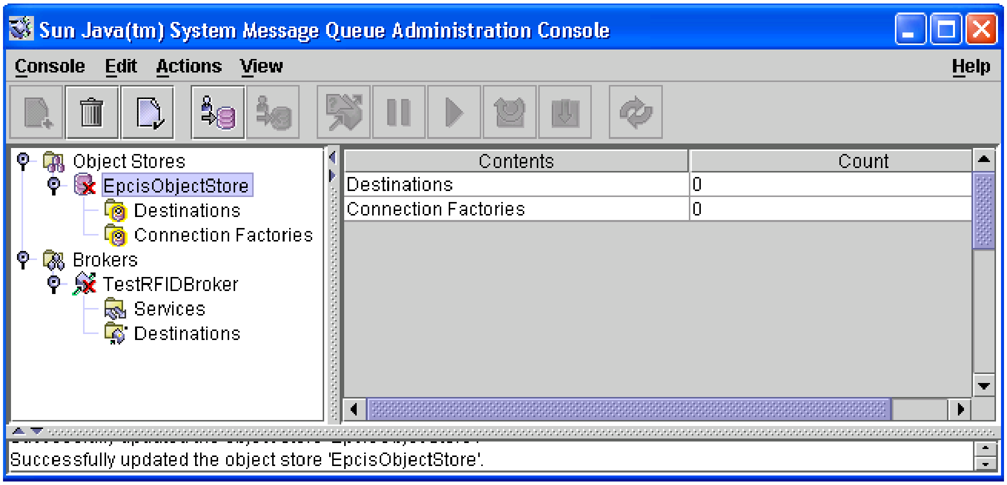
| Object Store Property Name | Example Value |
|---|---|
| `java.naming.factory.initial` | `com.sun.jndi.fscontext.RefFSContextFactory` |
| `java.naming.provider.url` | • (UNIX) – `file:///opt/imq/imq_admin_objects`<br>• (Microsoft Windows) – `file:///C:/Sun/Appserver/imq/imqdata` |
| `java.naming.security.prinicpal` | `tester` (or a user name that you specify) |
| `java.naming.security.credentials` | password for the user specified in `java.naming.security.principal` |

4. **In the Object Store Label field, type a name for this object store.**

   For this example, type `EpcisObjectStore`.

5. **When finished, click OK.**

   The new Object Store appears in the navigation tree.



| Contents | Count |
|---|---|
| Destinations | 0 |
| Connection Factories | 0 |

Successfully updated the object store 'EpcisObjectStore'.

6. **Right-click the new Object Store node in the navigation tree and choose Connect to Object Store.**

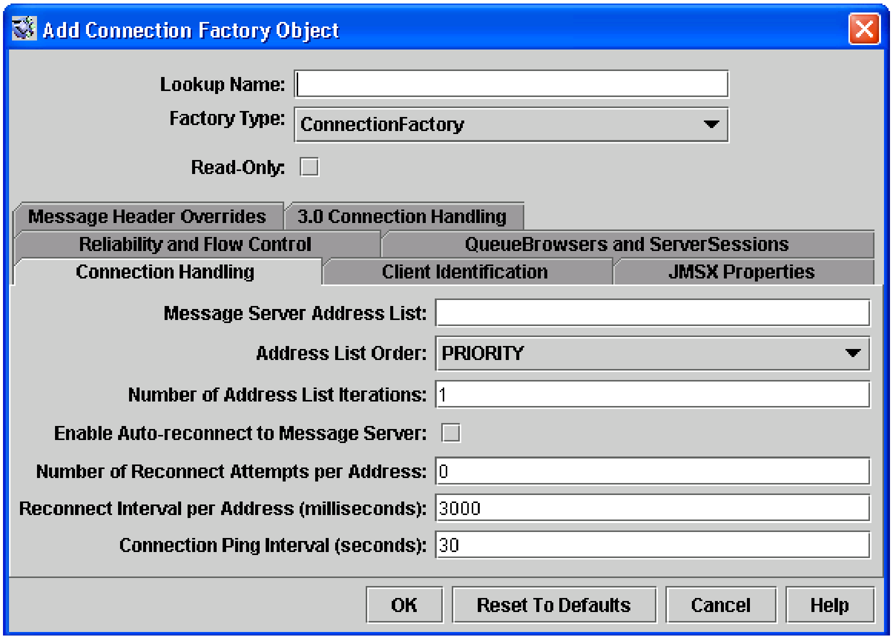
   You should see a successful connection.

## ▼ To Add a Topic Connection Factory

From the Message Queue admin console, add a new connection factory for the `epcisTopic` by following these steps:

1. **Select the Connection Factories node in the navigation tree.**
2. **From the menu, choose Actions → Add Connection Factory Object.**

   The Add Connection Factory Object dialog box appears.



Lookup Name:

Factory Type: ConnectionFactory

Read-Only: □

Message Server Address List:

Address List Order: PRIORITY

Number of Address List Iterations: 1

Enable Auto-reconnect to Message Server: □

Number of Reconnect Attempts per Address: 0

Reconnect Interval per Address (milliseconds): 3000

Connection Ping Interval (seconds): 30

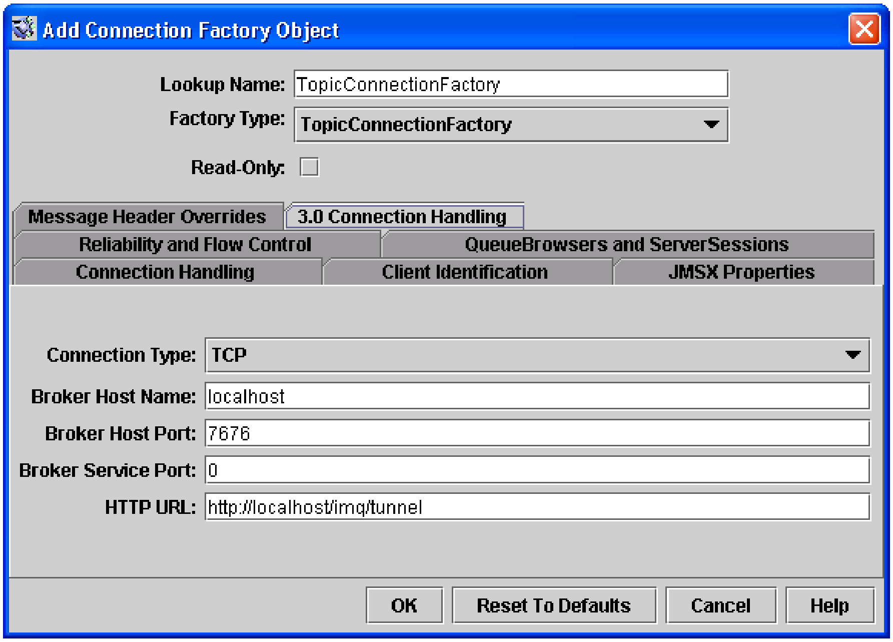
3. **Type the Lookup Name.**

   For this example, use `TopicConnectionFactory`.

4. **In the Factory Type field, select** `TopicConnectionFactory` **from the drop-down list.**

5. **Select the 3.0 Connection Handling tab and complete the fields as shown in the following screen capture.**

   In the fields, Broker Host Name and HTTP URL, replace `localhost` with the name of the machine where your Message Queue broker resides.



Add Connection Factory Object

Lookup Name: TopicConnectionFactory

Factory Type: TopicConnectionFactory

Read-Only: □

Connection Type: TCP

Broker Host Name: localhost

Broker Host Port: 7676

Broker Service Port: 0

HTTP URL: http://localhost/imq/tunnel

6. **Click OK.**

   The topic connection factory object is added to the object store.

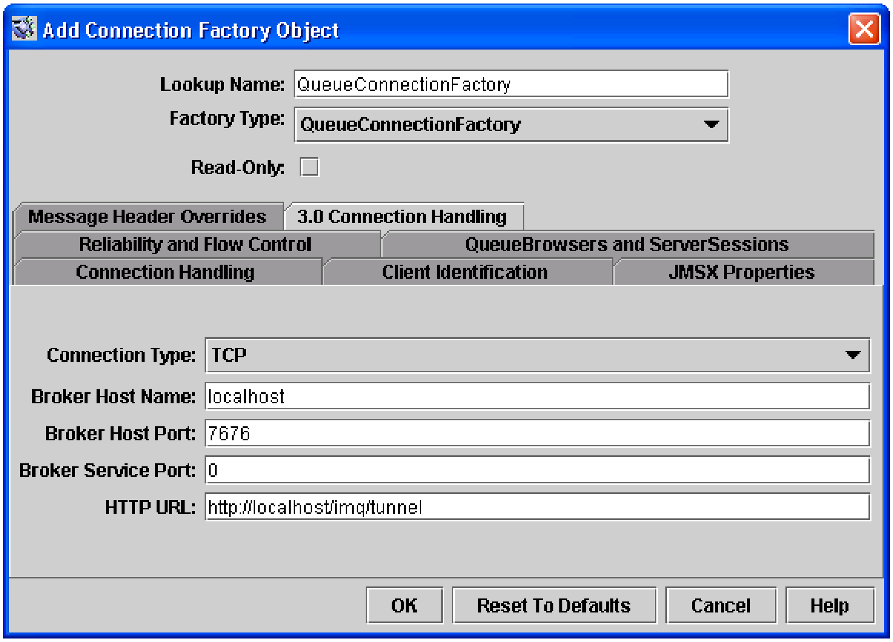
## ▼ To Add a Queue Connection Factory

1. **Select the Connection Factories node in the navigation tree.**
2. **From the menu, choose Actions → Add Connection Factory Object.**

   The Add Connection Factory Object dialog box appears.
3. **Type the Lookup Name.**

   For this example, use `QueueConnectionFactory`.
4. **In the Factory Type field, select `QueueConnectionFactory` from the drop-down list.**
5. **Select the 3.0 Connection Handling tab and complete the fields as shown in the following screen capture.**

   In the fields, Broker Host Name and HTTP URL, replace `localhost` with the name of the machine where your Message Queue broker resides.



Lookup Name: QueueConnectionFactory

Factory Type: QueueConnectionFactory

Read-Only: □

Connection Type: TCP

Broker Host Name: localhost

Broker Host Port: 7676

Broker Service Port: 0

HTTP URL: http://localhost/imq/tunnel

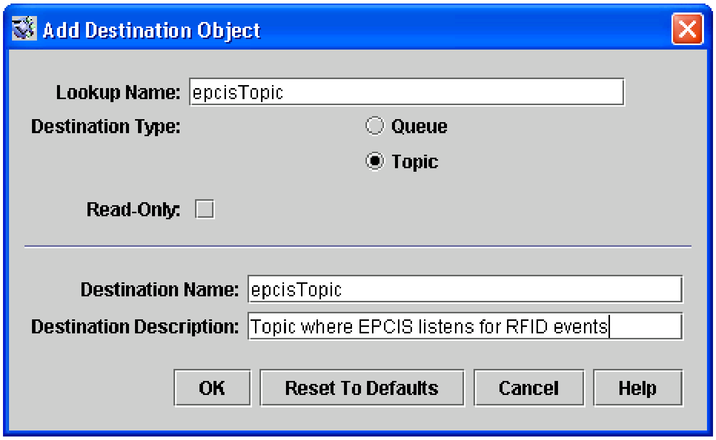
6. **Click OK.**

   The new queue connection factory object is added to the object store.

## ▼ To Add the EPCIS Destination Objects

1. **Select the Destinations node in the navigation tree.**
2. **From the menu, choose Actions → Add Destination Object.**

   The Add Destination Object dialog box appears.
3. **Add both a topic and a queue destination object as shown in the following screen captures.**



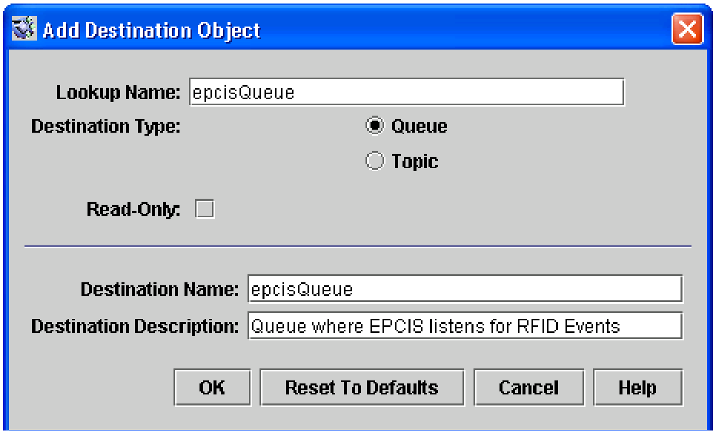
Lookup Name: epcisTopic

Destination Type: ○ Queue ● Topic

Read-Only: □

Destination Name: epcisTopic

Destination Description: Topic where EPCIS listens for RFID events



Lookup Name: epcisQueue

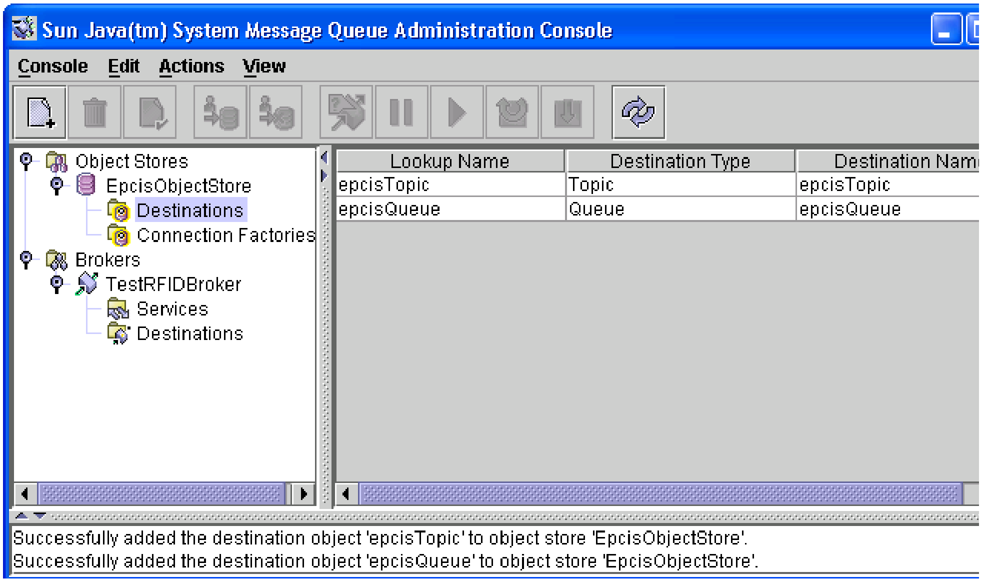
Destination Type: ● Queue ○ Topic

Read-Only: □

Destination Name: epcisQueue

Destination Description: Queue where EPCIS listens for RFID Events

4. **Confirm that your new object store appears similar to the following screen capture.**



| Lookup Name | Destination Type | Destination Nam |
|---|---|---|
| epcisTopic | Topic | epcisTopic |
| epcisQueue | Queue | epcisQueue |

```
Successfully added the destination object 'epcisTopic' to object store 'EpcisObjectStore'.
Successfully added the destination object 'epcisQueue' to object store 'EpcisObjectStore'.
```

5. **When you are finished, exit the Message Queue Admin Console.**

   You have completed the JMS configuration for your system. Now, you must configure the RFID Event Manager to send the tag events to the RFID Information Server. See "Configuring the RFID Event Manager to Use an EpcisJms Connector" on page 119.

# Configuring the RFID Event Manager to Use an EpcisJms Connector

The following procedures use the RFID Configuration Manager, a component of the RFID Event Manager. If you are unfamiliar with this component, review Chapter 2 to familiarize yourself with the GUI terminology and functionality before proceeding.

**Prerequisite** – Be sure that you have reviewed "Enabling Usage of JMS With the RFID Information Server" on page 106 and enabled JMS before starting this procedure.

Perform the following procedures:

- "To Define the EpcisJms Connector Role" on page 120
- "To Create the EpcisJms Configuration Object" on page 125
- "To Test the EpcisJms Connector" on page 126

## ▼ To Define the EpcisJms Connector Role

Use these steps to define a new role that uses the EpcisJms connector. A role specifies a series of filters and connectors that receive tag events and pass them to a *listener* application. In this example, the Westerner is the RFID Information Server.

Roles are described in Chapter 2 of this guide. Review "To Define RFID Event Manager Roles" on page 30 if you have not used the RFID Configuration Manager previously.

**Prerequisite** – You must create your JNDI object store before performing this procedure. See "Enabling Usage of JMS With the RFID Information Server" on page 106.

1. **Start the RFID Configuration Manager.**

   The location of the start script depends on your platform:

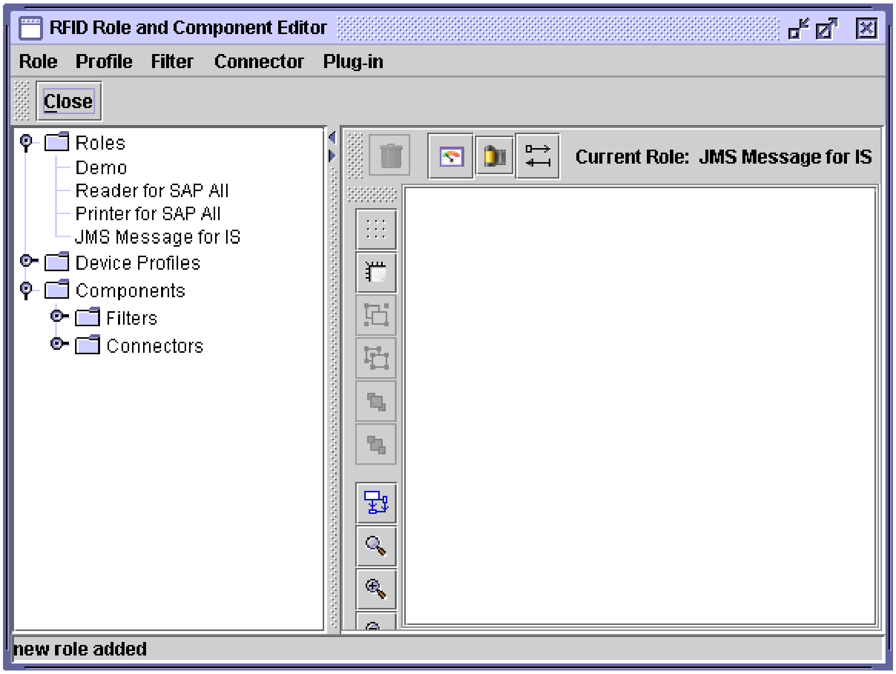
   - Solaris – /opt/SUNWrfid/bin
   - Linux – /opt/sun/rfidem/bin
   - Windows – Start → Choose Start → Programs → Sun Microsystems → Sun Java System RFID Software → Configuration Manager .

2. **Choose Roles → New.**

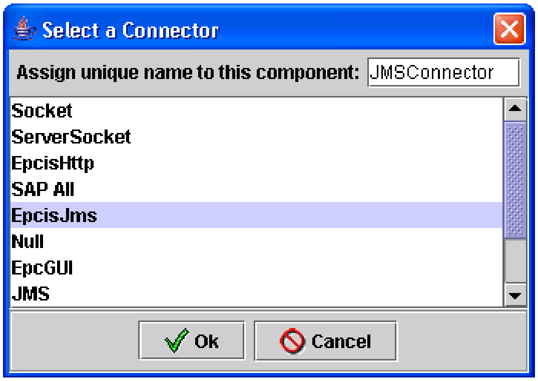
   The Select a Role dialog appears.

3. **Type the new role name and click Ok.**

   For this example, type JMS Message for IS. The drawing pane appears.

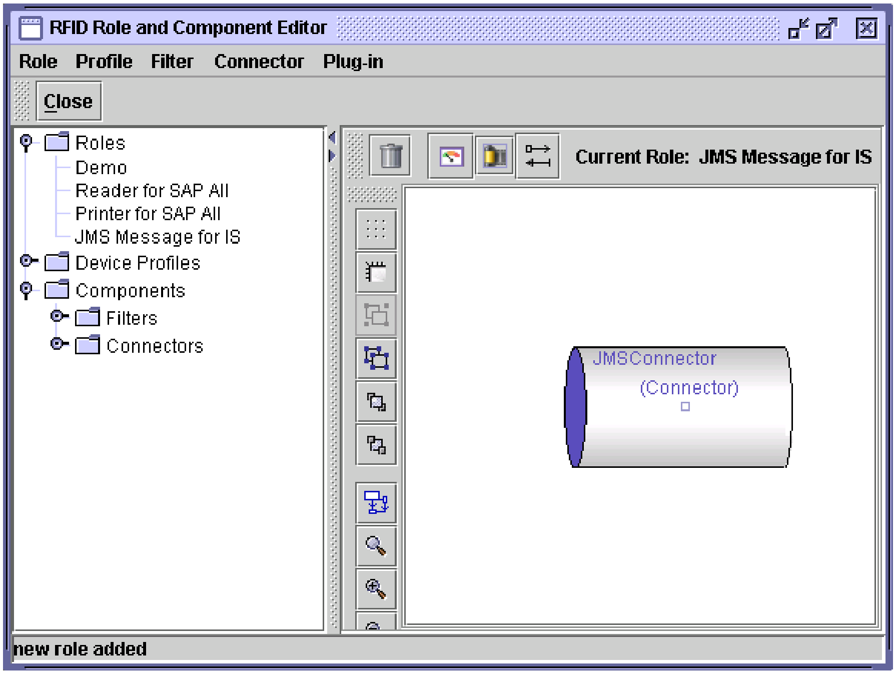


4. **Click the Add a connector icon (looks like a roll of film) and select `EpcisJms` from the list.**



5. **Type a name and click OK.**

   For the example, type `JMSConnector`. The `Epcis` connector named JMSConnector appears in the drawing pane.



new role added

6. **Right-click the connector (move the cursor over the small square at the center of the Connector symbol) and select Edit.**

   The Connector Details dialog appears.

7. **Confirm that the Configuration Properties fields contain the correct values for your JMS configuration and click OK.**

   The values of the `java.naming` properties must match the values in your JNDI object store as shown in the following table.

| Configuration Property Name | Value |
|---|---|
| `java.naming.factory.initial` | `com.sun.jndi.fscontext.RefFSContextFactory` |
| `java.naming.provider.url` | • UNIX – `file:///opt/imq/imq_admin_objects`<br>• Microsoft Windows – `file:///C:/Sun/Appserver/imq/imqdata` |
| `java.naming.security.prinicpal` | `tester` (the user name that you specified for connecting to the RFID Information Server) |
| `java.naming.security.credentials` | password for the user specified in `java.naming.security.principal` |
| `TopicName` | `epcisTopic` |
| `ConnectionFactory` | `TopicConnectionFactory` |

8. **Add a Delta filter by using the Add a filter icon (looks like a meter).**

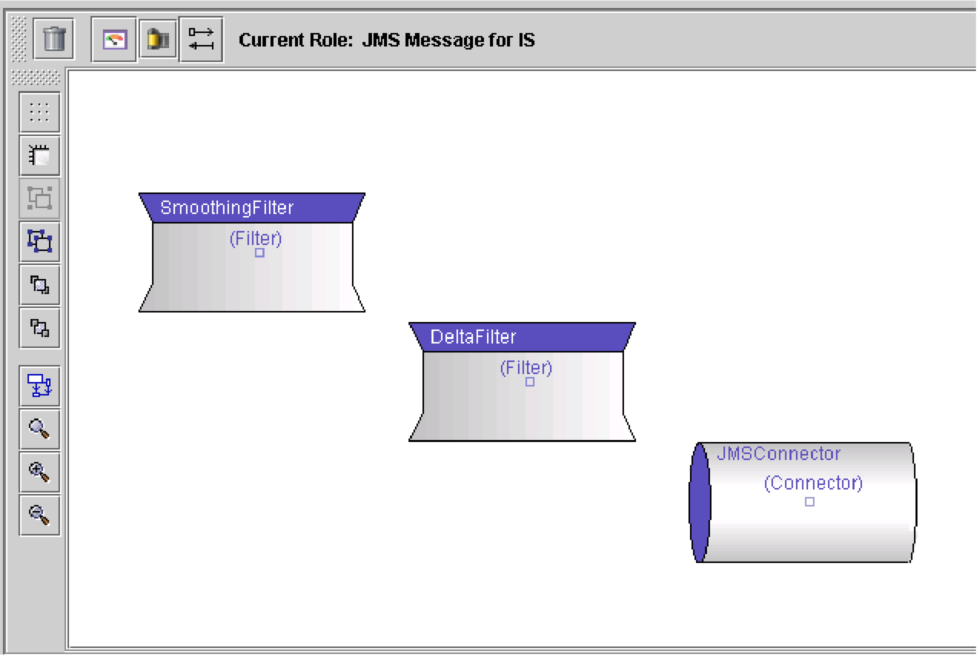
   You must use a Delta filter to feed the tag events to the `EpcisJms` connector. If you do not add the Delta filter to this role, the tag events are ignored by the connector.

9. **Type a name for your Delta filter and click OK.**

10. **Add a Smoothing filter by using the Add a filter icon (looks like a meter).**

11. **Type a name for your Smoothing filter and click OK.**

    The drawing pane appears similar to the following screen capture.



12. **Right-click on the Smoothing filter and select Create Input from the popup menu.**

    A new input object appears on the drawing pane.

13. **To connect everything together, click the port (small square) at the center of the input and drag it to the Smoothing filter.**

14. **Click the port of the Smoothing filter and drag it to the Delta filter.**

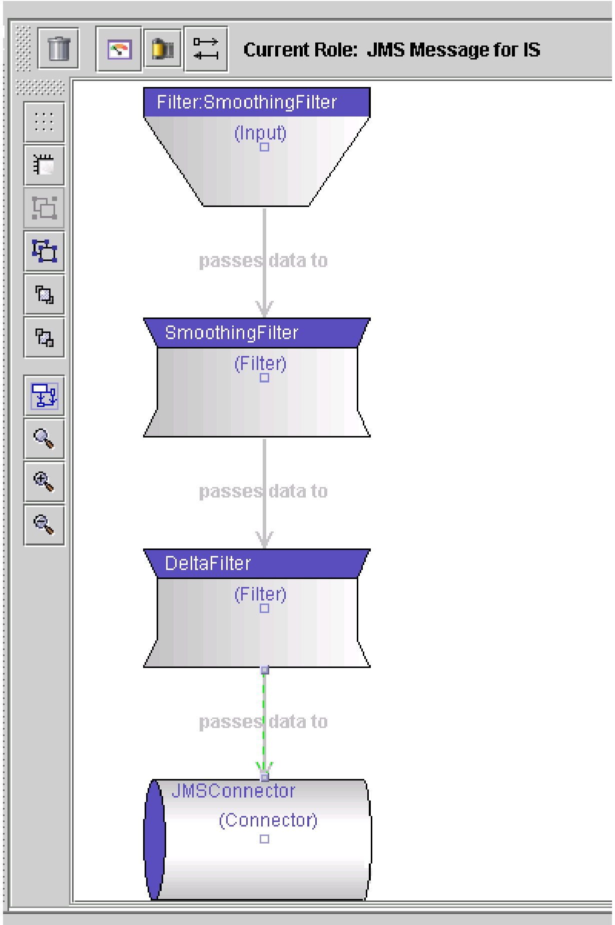
15. **Click the port of the Delta filter and drag it to the JMSConnector.**

16. **Click on the Auto-Layout icon to arrange the components vertically on the drawing pane.**

    This enables you to see how the tag events flow through this role.

**Note –** The drawing pane icons are described in TABLE 2-1.

Your drawing pane appears similar to the following screen capture.



17. **Click Close.**

    Your work is not yet saved. To save your work at this time, choose File → Save from the RFID Configuration Manager main menu.

18. **You must complete the following procedure to configure the** `EpcisJMS` **configuration object.**

## ▼ To Create the `EpcisJms` Configuration Object

1. **Choose Configuration → New.**

   The Select the Base Role dialog box appears.

2. **Select the role, JMS Message for IS, that you defined in the previous procedure and click OK.**

   The Configuration Object properties dialog appears. Notice that there are three tabs corresponding to the components that you defined in the JMS Message for IS role.

3. **Type a name in the Configuration Object Name field.**

   For this example, type `JMSConfigObject`.

4. **To configure the input point for the configuration object, select a reader by following these steps:**

   a. **Click Select a Reader.**

   b. **(For demonstration or verification purposes) Select PMLReader.**

      When you select the reader, the configuration properties for that reader profile appear.

   c. **(If using a real reader, rather than the PMLReader) Select your reader and confirm that the reader properties correspond to the reader you are using.**

      When defining the `EpcisJms` connector in a real-world situation, you would define your readers (devices) first. Then, when you define your configuration objects, the list of possible input points would display all your defined readers (devices). See Chapter 2 for more information.

5. **Click OK.**

6. **If the drawing pane displays the default Demo configuration object, delete it following these steps.**

   a. **Select the Demo configuration object.**

   b. **Choose Configuration → Delete.**

7. **Choose File > Save to save your changes and click OK when prompted.**

8. **(Optional) To exit the RFID Configuration Manager, choose File → Exit.**

9. **(Optional) To test the `EpcisJms` connector, perform the next procedure "To Test the `EpcisJms` Connector" on page 126.**

## ▼ To Test the EpcisJms Connector

1. **Start or restart your application server.**
2. **Start or restart the RFID Event Manager.**

   Each time you change a configuration object, the RFID Event Manager must be restarted in order for the changes to take effect. Wait until the system is completely started or you may lose tag events. Depending on the components you used in your role, the RFID Tag Viewer may or may not appear (it depends on the objects in your role). The delay lets the system come up completely before generating EPC tags.
3. **Start the PML Reader.**
4. **Start the RFID Tag Viewer from the command line using the following options.**

   ```
   tagviewer -p PMLReader
   ```

   You should see tags appearing in the RFID Tag Viewer.
5. **After a few seconds, you can stop the PML Reader.**
6. **(Optional) To verify that events are flowing from the RFID Event Manager to the RFID Information Server, follow these steps:**

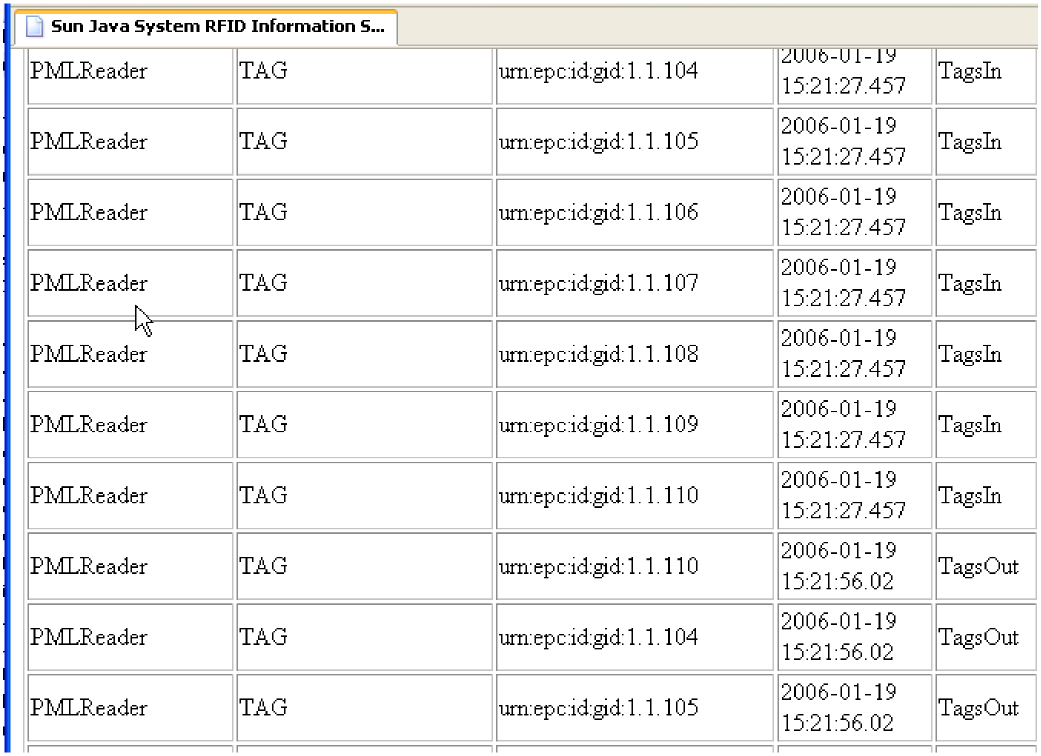
   a. **In your web browser, access the RFID Information Server index page by typing the EPCIS URL.**

      For example, `http://localhost/epcis.`

   b. **Select Epcis Reporting Framework → Epcis Tables.**

      Scroll down to the OBSERVATION_LOG table.

      The TagsIn and TagsOut that have been added to the OBSERVATION_LOG table indicate that tags were read and were taken out of view from the RFID Event Manager. You should see something similar to the following screen capture.



| | | | | |
|---|---|---|---|---|
| PMLReader | TAG | urn:epc:id:gid:1.1.104 | 2006-01-19 15:21:27.457 | TagsIn |
| PMLReader | TAG | urn:epc:id:gid:1.1.105 | 2006-01-19 15:21:27.457 | TagsIn |
| PMLReader | TAG | urn:epc:id:gid:1.1.106 | 2006-01-19 15:21:27.457 | TagsIn |
| PMLReader | TAG | urn:epc:id:gid:1.1.107 | 2006-01-19 15:21:27.457 | TagsIn |
| PMLReader | TAG | urn:epc:id:gid:1.1.108 | 2006-01-19 15:21:27.457 | TagsIn |
| PMLReader | TAG | urn:epc:id:gid:1.1.109 | 2006-01-19 15:21:27.457 | TagsIn |
| PMLReader | TAG | urn:epc:id:gid:1.1.110 | 2006-01-19 15:21:27.457 | TagsIn |
| PMLReader | TAG | urn:epc:id:gid:1.1.110 | 2006-01-19 15:21:56.02 | TagsOut |
| PMLReader | TAG | urn:epc:id:gid:1.1.104 | 2006-01-19 15:21:56.02 | TagsOut |
| PMLReader | TAG | urn:epc:id:gid:1.1.105 | 2006-01-19 15:21:56.02 | TagsOut |

## Configuring the RFID Event Manager to Use an EpcisHttp Connector

To use HTTP to communicate with the RFID Information Server, use an EpcisHttp connector. The tasks for configuring an EpcisHttp connector are similar to configuring a EpcisJms connector, except you do not need to do the JMS configuration. Perform the following procedures:

- "To Define the Epcis Connector Role" on page 128
- "To Create the EpcisHttp Configuration Object" on page 130
- "To Test the EpcisHttp Connector" on page 131

## ▼ To Define the Epcis Connector Role

1. **Start the RFID Configuration Manager.**

2. **Choose Roles → New.**

   The Select a Role dialog appears.

3. **Type the new role name and click Ok.**

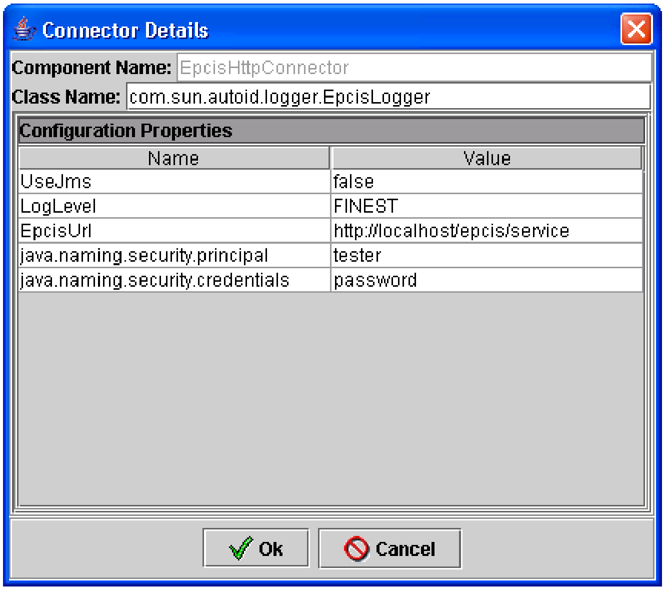
   For this example, type `testEpcisHttpRole`.

4. **Click the Add a connector icon (looks like a roll of film) and select `EpcisHttp` from the list.**

5. **Type a name and click OK.**

6. **To edit/confirm the properties, right-click the connector and choose Edit.**

   Move the cursor over the port (the small square at the center of the connector object). The Connector Details dialog box appears.



Component Name: EpcisHttpConnector

Class Name: com.sun.autoid.logger.EpcisLogger

Configuration Properties

| Name | Value |
|---|---|
| UseJms | false |
| LogLevel | FINEST |
| EpcisUrl | http://localhost/epcis/service |
| java.naming.security.principal | tester |
| java.naming.security.credentials | password |

7. **Confirm that the properties contain the correct values, and click OK.**

   - Confirm the `EpcisUrl` is correct for your RFID Information Server.
   - Confirm that the `java.naming.security.principal` value contains the correct user name for accessing the RFID Information Server.
   - Confirm that the `java.naming.security.credentials` value contains the correct password for the user specified in `java.naming.security.principal`.

8. **Add a Smoothing filter and a Delta filter.**

   See Step 8 through Step 11 in "To Define the EpcisJms Connector Role" on page 120.

9. **Right-click the Smoothing filter and choose Create Input from the context menu.**

10. **To connect everything together, click the port (small square) at the center of the input and drag it to the Smoothing filter.**

11. **Click the port of the Smoothing filter and drag it to the Delta filter.**

12. **Click the port of the Delta filter and drag it to the** `EpcisHttpConnector`**.**

13. **Drag the components to arrange them on the drawing pane as needed or click on the Auto-Layout icon to arrange the components vertically.**

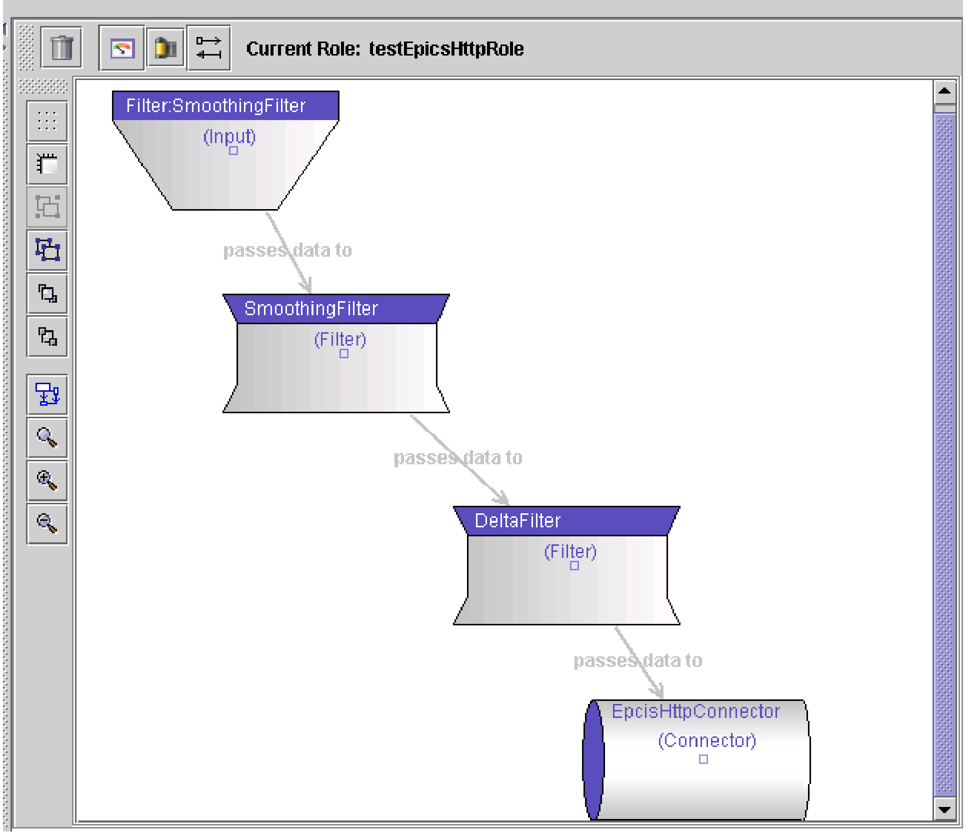
    This enables you to see how the tag events flow through the role.

---

**Note –** The drawing pane icons are described in TABLE 2-1.

---

Your drawing pane appears similar to the following screen capture.



14. **Click close and proceed to the next procedure,**

    Your work is not yet saved. Do not exit the RFID Configuration Manager without saving your work.

## ▼ To Create the `EpcisHttp` Configuration Object

1. **Start the RFID Configuration Manager.**
2. **If the drawing pane displays the default Demo configuration object, delete it following these steps.**

    a. **Select the Demo configuration object.**

    b. **Choose Configuration → Delete.**

3. **Choose Configuration → New.**

    The Select the Base Role dialog box appears.

4. **For this example, select the role, `testEpcisHttpRole`, that you defined in the previous procedure and click OK.**

   The Configuration Object properties dialog appears. Notice that there are three tabs corresponding to the components that you defined in the role.

5. **Type a name in the Configuration Object Name field.**

   For this example, type `HTTPConfigObject`.

6. **To configure the input point for the configuration object, select a reader by following these steps:**

   a. **Click Select a Reader.**

   b. **(For demonstration or verification purposes) Select PMLReader.**

      When you select the reader, the configuration properties for that reader profile appear.

   c. **(If using a real reader, rather than the PMLReader) Select your reader and confirm that the reader properties correspond to the reader you are using.**

      When defining the `EpcisHttp` connector in a real-world situation, you would define your readers (devices) first. Then, when you define your configuration objects, the list of possible input points would display all your defined readers (devices). See Chapter 2 for more information.

7. **Click OK.**

8. **Choose File > Save to save your changes and click OK when prompted.**

9. **(Optional) To exit the RFID Configuration Manager, choose File → Exit.**

10. **(Optional) To test the `EpcisHttp` connector, perform the next procedure**

## ▼ To Test the `EpcisHttp` Connector

1. **Start or restart the RFID Event Manager.**

   Each time you change a configuration object, the RFID Event Manager must be restarted in order for the changes to take effect.

   Wait until the system is completely started or you may lose tag events. Depending on the components you used in your role, the RFID Tag Viewer may or may not appear (it depends on the objects in your role). The delay lets the system come up completely before generating EPC tags.

2. **Start the PML Reader.**

3. **Start the RFID Tag Viewer from the command line using the following options.**

   ```
   tagviewer -p PMLReader
   ```

   You should see tags appearing in the RFID Tag Viewer.

4. **After a few seconds, you can stop the PML Reader.**

5. **(Optional) To verify that events are flowing from the RFID Event Manager to the RFID Information Server, follow these steps:**

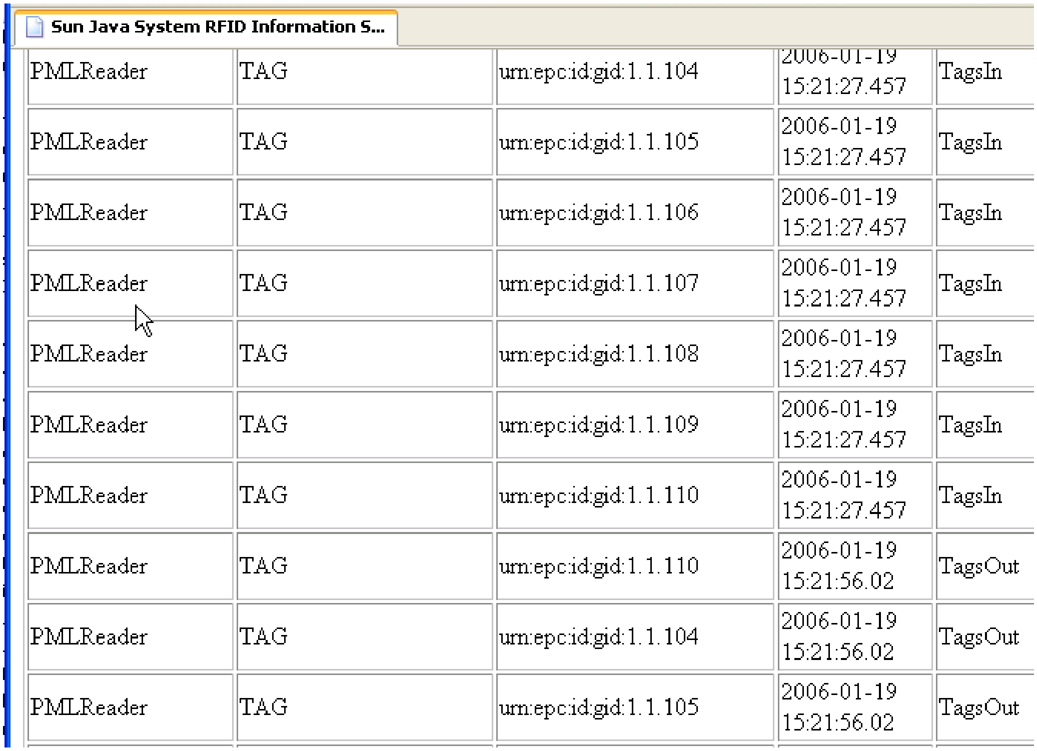
   a. **In your web browser, access the RFID Information Server index page by typing the EPCIS URL.**

   For example, http://localhost/epcis.

   b. **Select Epcis Reporting Framework → Epcis Tables.**

   Scroll down to the OBSERVATION_LOG table.

   The TagsIn and TagsOut that have been added to the OBSERVATION_LOG table indicate that tags were read and were taken out of view from the RFID Event Manager. You should see something similar to the following screen capture.



| | | | | |
|---|---|---|---|---|
| PMLReader | TAG | urn:epc:id:gid:1.1.104 | 2006-01-19 15:21:27.457 | TagsIn |
| PMLReader | TAG | urn:epc:id:gid:1.1.105 | 2006-01-19 15:21:27.457 | TagsIn |
| PMLReader | TAG | urn:epc:id:gid:1.1.106 | 2006-01-19 15:21:27.457 | TagsIn |
| PMLReader | TAG | urn:epc:id:gid:1.1.107 | 2006-01-19 15:21:27.457 | TagsIn |
| PMLReader | TAG | urn:epc:id:gid:1.1.108 | 2006-01-19 15:21:27.457 | TagsIn |
| PMLReader | TAG | urn:epc:id:gid:1.1.109 | 2006-01-19 15:21:27.457 | TagsIn |
| PMLReader | TAG | urn:epc:id:gid:1.1.110 | 2006-01-19 15:21:27.457 | TagsIn |
| PMLReader | TAG | urn:epc:id:gid:1.1.110 | 2006-01-19 15:21:56.02 | TagsOut |
| PMLReader | TAG | urn:epc:id:gid:1.1.104 | 2006-01-19 15:21:56.02 | TagsOut |
| PMLReader | TAG | urn:epc:id:gid:1.1.105 | 2006-01-19 15:21:56.02 | TagsOut |

CHAPTER 7

# Sun Java System RFID Information Server Configuration

The Sun Java System RFID Information Server (RFID Information Server) is a Java 2 Platform, Enterprise Edition (J2EE™) application that serves as an interface for the storage and query of the tag data captured by RFID readers. The RFID Information Server is typically used to translate a set of low-level observations into higher-level business functions.

This chapter includes the following topics:

- RFID Information Server Architecture and Database Overview
- RFID Database Information Tables
- Configuring the RFID Information Server Database
- Configuring RFID Information Server Clients
- Securing the RFID Information Server

## RFID Information Server Architecture and Database Overview

The Sun Java System RFID Information Server is a J2EE application that serves as an interface for capture and query of EPC-related and non-EPC tag data. As outlined in "Information Tables," EPC-related tag data can include RFID tag observation data from event managers as well as information that maps the EPCs to higher-level business data. The RFID Information Server is typically used to translate a set of low-level observations into higher-level business functions.

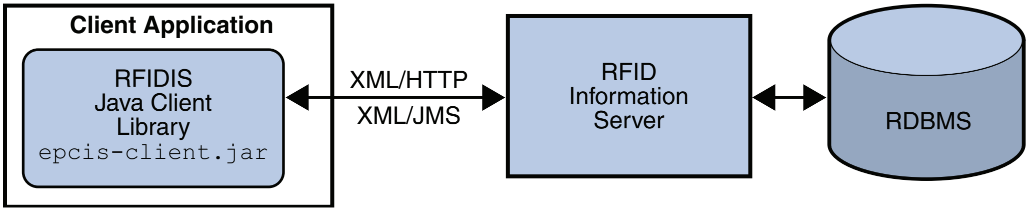


**FIGURE 7-1** Sun Java System RFID Information Server

Alternatively, you can consider the EPC-related tag data as being either static or timestamped. Static data includes the serial-level and product-level properties of tagged items, such as the GTIN number of a box of detergent. Timestamped data is collected over a period of time and typically changes more often than static data. An example of timestamped data is the relationship between an EPC and a business transaction, such as an advance shipment notice.

The RFID Information Server is deployed on one of the supported J2EE-compliant application servers. Other applications can interface with the RFID Information Server through XML message exchange. The RFID Information Server also supports HTTP and Java Message Service (JMS) API message transports. The RFID Information Server stores all data in a relational database management system (RDBMS). The RFID Information Server has been tested with Oracle 9i Database, Oracle 10g Database, and PostgreSQL 8.0.4.

---

**Note –** EPCGlobal has recently chartered a working group to define the specifications for an EPC Information Server (EPCIS). Sun actively supports and monitors the activities of the working group and participates in the endorsement the specifications as they become final. EPCIS is the industry term for middleware such as the Sun Java System RFID Information Server.

---

# RFID Database Information Tables

The RFID Information Server stores EPC-related information in database tables that include the following:

- **Product-level and serial-level data –** This data is captured in the ORGANIZATION, PRODUCT and UNIT tables.

  These tables have a hierarchical relationship. Each PRODUCT is linked to an ORGANIZATION through the ORGANIZATION_ID attribute. Similarly, each UNIT is linked to a PRODUCT through the PRODUCT_ID attribute.

- **Sensors –** The SENSOR table lists all the RFID readers and antennas in use in the RFID sensor network.
- **Tag observations –** Observations are generated by RFID readers and captured in two tables in the RFID Information Server.

  The CURRENT_OBSERVATION table captures the output of a Delta filter. The purpose of this filter is to capture the currently visible EPC tags at any particular sensor.

  The OBSERVATION_LOG table captures the history of all tag observations.
- **Containment –** Containment refers to the aggregation of one or more tagged items into a larger item. Relationships between a container and its individual contents are captured in the CONTAINMENT table. The parent EPC identifies the container. One or more child EPCs identify the contents. A child EPC can in turn be the EPC of a container, so the containment relationship can be arbitrarily deep.
- **Transaction IDs –** The TX_LOG table maps a set of EPCs to a business transaction ID similar to a purchase order number or an advance shipment notification number.
- **Tag allocation –** In a distributed information service framework, a central RFID Information Server can assign ranges of EPC numbers to local RFID Information Servers or applications at distribution centers. Each local RFID Information Server requests a range of serialized EPCs from a given manager/object class. The central RFID Information Server guarantees that each EPC range is used only by the requesting RFID Information Server. The RFID Information Servers at the distribution centers track the EPC ranges allocated to and deallocated from the clients.

## RFID Information Server Database Scripts

Refer to the *Sun Java System RFID Software 3.0 Installation Guide* for details on setting up sample databases and using the database scripts that are available with the product.

Refer to the *Sun Java System RFID Software 3.0 Developer's Guide* for API information on accessing the RFID Software databases.

# Configuring the RFID Information Server Database

This section describes RFID Information Server database configuration information. The following topics are described:

- Database Schema Description
- Configuring the Logging Parameters

## Database Schema Description

The `EpcisDbSchema.xml` file describes the database schema in a database-independent manner. This schema is used by the client and the server to perform type checking and data validation at run time. This file also specifies extended attributes for the UNIT, PRODUCT and ORGANIZATION tables outlined in the section "RFID Database Information Tables" on page 134.

All `<table>` elements have a fixed attribute section that lists the columns in the table. The UNIT, PRODUCT and ORGANIZATION table elements also have an extended attribute section.

The `EpcisDbSchema.xml` file is located in the document root of the J2EE application. For example, for a default installation on Solaris OS, the document root of the RFID Information Server is as follows:

- Sun Java System Application Server Platform Edition 8.1 2005Q2 UR2 – `/opt/SUNWappserver/domains/domain1/applications/j2ee-apps/epcis/sun-rfid-is_war`
- Sun Java System Application Server Enterprise Edition 8.1 2005Q1 – `/opt/SUNWappserver/domains/domain1/applications/j2ee-apps/epcis/sun-rfid-is_war`

**Note –** The application server installation path varies depending on the version and platform that you are using.

## ▼ To Modify the RFID Information Server Database Schema File

1. **Save a copy of `EpcisDbSchema.xml` in a permanent location.**
2. **Modify the file to add or delete extended attributes.**
3. **Copy the modified file to the document root of the application server.**
4. **Restart the application server.**

---

**Caution –** The `EpcisDbSchema.xml` file is intended to be modified once at deployment. If changes are made to the file after the database has been populated, the result might be inconsistent data in the database.

---

### Example for a `<table>` Element

In this example the PRODUCT table has one extended attribute, SIZE_OZ. The attribute is of type `float`.

```
<!-- Insert extended attributes below -->
<!-- Type is always String -->
<!-- paramNum is always 3 -->
<attribute extended="true">
    <name>SIZE_OZ</name>
    <dbName>SIZE_OZ</dbName>
    <type>FLOAT</type>
    <paramNum>3</paramNum>
</attriute>
</table>
```

### Adding Extended Attributes to a Table

The following shows the template for an extended attribute.

```
<attribute extended="true">
<name></name>
<dbName></dbName>
<type></type>
<paramNum>3</paramNum>
</attribute>
```

### ▼ To Add Extended Attributes Using the Template

1. **Enter the name of the attribute in the** `name` **and** `dbName` **tags.**
2. **Enter the type of the attribute. The `type` can be one of:**

   `STRING`, `FLOAT`, `LONG`, `TIMESTAMP`, `INTEGER` or `BOOLEAN`

   The `paramNum` tag is always 3.

## Configuring the Logging Parameters

The logging parameters are set using Application Server's administration interface. Refer to the Application Server administration guide for more details.

In general, the logging service is an element within the J2EE service element category in the `server.xml` file, as described in the Sun Java System Application Server Configuration File Reference. The log service is used to configure the system logging service, which can include the following log files:

- Server log
- Access log
- Transaction log
- Virtual server log

Configuration of the system logging service can include specifying values for various attributes of the log service element. Using Application Server's administration interface, you can configure any of the following attributes for the log service element:

- Log File
- Default Log Level
- Log Standard Out content to event log
- Log Standard Error content to event log
- Echo to Standard Error
- Create Console
- Log Virtual Server ID
- Write to System Log

For general usage with production systems, reset the Default Log Level to WARNING. For debugging purposes set the Log Level to INFO or finer granularity.

# Configuring RFID Information Server Clients

The RFID Information Server client library (shown in FIGURE 7-1 as "EPCIS Java Client Library") requires two Java system properties.

- `rfidis.db.schema` – Specifies the HTTP URL to the `EpcisDbSchema.xml` file
- `java.util.logging.config.file` – Specifies the path to the client logging properties file

## ▼ To Add the Java System Properties

1. **From the Application Server Admin Console, choose Application Server → JVM Settings → JVM Options.**
2. **Click Add JVM Option.**
3. **Type the following option value:**

```
-Drfidis.db.schema=http://localhost/epcis/Epcis/DbSchema.xml
```

4. **Press Enter.**
5. **Click Add JVM Option.**
6. **Type the following option value:**

```
-Djava.util.logging.config.file=
http://localhost/epcis/logging.properties
```

7. **Press Enter**
8. **Click Save.**

# Securing the RFID Information Server

Using the Application Server Admin Console, you can manage user access to the RFID Information Server. Basic user authentication is the first step to securing the RFID Information Server. Component level security encompasses web components and EJB components. A secure web container authenticates users and authorizes access to a servlet or JSP by using the security policy defined in the servlet XML deployment descriptors (web.xml and sun-web.xml files).

For more detailed information on writing secure applications, see the documentation for your application server as follows:

- *Sun Java System Application Server Platform Edition 8.1 2005Q2 Update 2 Developer's Guide* at http://docs.sun.com/app/docs/doc/819-2642
- *Sun Java System Application Server Enterprise Edition 8.1 2005Q2 Developer's Guide* at http://docs.sun.com/app/docs/doc/819-2556

For the RFID Information Server, two roles are defined in WEB-INF/web.xml: readonly and readwrite. The users with the readonly role can only find tags from RFID Information Server. The users with the readerwrite role can also modify the RFID Information Server database for tags. The default realm used is the file realm. The realm can be changed to ldap or other supported realms.

Two role-mappings are predefined in WEB-INF/sun-aplication.xml. Where, a principal user *guest* and a group *viewer* are mapped to the role readonly. A principal user *tester* and a group *modifier* are mapped to the role readwrite. Using this definition, you can use the Application Server Admin Console to add a new user or assign existing users to group *viewer* or *modifier* in file realm for corresponding access privilege.

For more information on defining users for the RFID Information Server, see the *Sun Java System RFID Software 3.0 Installation Guide*, Chapter 5, section Defining Valid Users for RFID Information Server.

APPENDIX **A**

# RFID Device Adapter Reference

This appendix describes the supported RFID device profiles (also known as adapters) and their properties. This information can be seen and modified using the RFID Configuration Manager. See "Managing Device Profiles" on page 46 for procedures.

The following tables list the properties (also known as attributes), a description and the valid values. Additional descriptive information for some properties follows the table where necessary to expand on the property usage. The properties that are common to all device profiles are described in detail in the section "Device Profiles for Supported RFID Devices" on page 47. The complete list of supported readers and other devices described in TABLE 2-3 found in Chapter 2

## Adapter for AWID Readers

This adapter is used to communicate with the AWID MPR-2010 reader. The adapter properties are defined in the following table.

**TABLE A-1** Adapter Properties for the AWID MPR-2010 Reader

| Property Name | Description | Values |
|---|---|---|
| Name | Unique name identifying this adapter | Sample value = AWIDAdapter |
| `classname` | Java class name. | `com.sun.autoid.adapter.awid.AWID2010Adapter` |

**TABLE A-1** Adapter Properties for the AWID MPR-2010 Reader *(Continued)*

| Property Name | Description | Values |
|---|---|---|
| **Common Properties** | See TABLE 2-4 for general description and values. | |
| LogLevel<br>hostname<br>port<br>readerepc<br>autoread<br>communicationTimeout<br>scanDuration | | hostname – Must be a static IP address or specified to boot from DHCP and obtain the IP address specified by the value of this property<br>port – default value = 4000 |
| **Additional Properties** | | |
| readCommandType | The AWID reader can read tags in multiple modes. | The values are:<br>• PORTAL_IDs<br>• IDs<br>• IDs_WITH_SELECT<br>• SINGLE_TAG_METER<br>• READ_SINGLE_TAGID<br>• READ_SINGLE_TAGID_TIMEOUT |
| tagType | The reader can read different type of tags. Specifying more than one property would set it to Multi-Protocol mode | A comma-separated list of one or more of the following values:<br>• EPC_CLASS_0<br>• EPC_CLASS_1<br>• INTERMEC<br>• EM<br>• EPC_CLASS_1.19<br>• MULTI_PROTOCOL |
| EM_Version | Required when reading in IDs mode for tagType set to EM | • EM_4022 (default)<br>• EM_4222 |

**TABLE A-1** Adapter Properties for the AWID MPR-2010 Reader *(Continued)*

| Property Name | Description | Values |
|---|---|---|
| writeEPCRetry | Number of tries to write before returning. | 1 – Default = 1<br>0 – Write until complete |
| sleeptime | The sleep time between reads in milliseconds. | Default = 250 msec |
| gatherUserData | If true, reads the user data in the transponder (RFID tag) and associates it with the identifier during the inventory round. This information is then passed along the processing chain to filters and connectors, which in turn interpret the user data appropriately. | false (default)<br>true |

# Adapter for Feig LRU1000 Readers

This adapter is used to communicate with the Feig LRU1000 reader. The adapter properties are shown in the following table.

**TABLE A-2** Adapter Properties for the Feig LRU1000 Readers

| Property Name | Description | Values |
|---|---|---|
| Name | Unique name identifying this adapter. | Sample value = FeigLRU1000Adapter |
| classname | Java class name.. | com.sun.autoid.adapter.feig.lru1000.LRU1000Adapter |
| **Common Properties** | See TABLE 2-4 for general description and values | |
| LogLevel<br>hostname<br>port<br>readerepc<br>autoread<br>communicationTimeout<br>scanDuration | | |

**TABLE A-2** Adapter Properties for the Feig LRU1000 Readers *(Continued)*

| Property Name | Description | Values |
|---|---|---|
| **Additional Properties** | . | |
| `antennaSequence` | A list of which antenna ports have antennas connected. | One or more comma-separated port numbers. For example, 1,2,4 |
| `tagIdRepresentation` | Specifies the encoding to use for interpreting the ID on the tag. | Values are:<br>• ISO<br>• EPC<br>Default value is EPC. |
| `protocolType` | Specifies which air interface protocol to activate. Only tags of the specified typ are reported. | Values are:<br>• ISO<br>• EPC1G1<br>• EPC1G2 |

# Adapter for Feig ISCMR100/PR100 Readers

This adapter is used to communicate with the Feig Electronic ID ISC MR.100 and Feig Electronic ID ISC PR.100 readers. The adapter properties are shown in the following table.

**TABLE A-3** Adapter Properties for the Feig ISCMR100/PR100 Readers

| Property Name | Description | Values |
|---|---|---|
| `Name` | Unique name identifying this adapter. | Sample value = `ISCMR100FeigReader` |
| `classname` | Java class name. | `com.sun.autoid.adapter.feig.obidiscan.ISCMR100Adapter` |
| **Common Properties** | See TABLE 2-4 for general description and values | |
| `LogLevel` | | |
| `hostname` | | Must be configured with a static IP address, or to boot from DHCP and obtain the IP address specified in this property. |
| `port` | | |

**TABLE A-3** Adapter Properties for the Feig ISCMR100/PR100 Readers *(Continued)*

| Property Name | Description | Values |
|---|---|---|
| readerepc | | |
| autoread | | |
| communicationTimeout | | |
| scanDuration | | |

# Adapter for Intermec IF5 Readers

This adapter is used to communicate with Intermec Intellitag IF5 readers. The adapter properties are shown in the following table.

**TABLE A-4** Adapter Properties for the Intermec Intelligtag IF5 Reader

| Property Name | Description | Values |
|---|---|---|
| Name | Unique name identifying this adapter. | Sample value = IntermecIF5_Adapter |
| classname | Java class name. | com.sun.autoid.adapter.intermec.IntermecIF5_Adapter |
| **Common Properties** | See TABLE 2-4 for general description and values | |
| LogLevel<br>hostname<br>port<br>readerepc | | |
| autoread | | Default value = false |
| communicationTimeout | | Default value = 10000 msec |

**TABLE A-4** Adapter Properties for the Intermec Intelligtag IF5 Reader *(Continued)*

| Property Name | Description | Values |
|---|---|---|
| scanDuration | | Default value is 500 msec |
| **Additional Properties** | | |
| tagType | The type of transponder (RFID tag) the reader expects.<br>Only one value is valid at a time. | The valid types are the following:<br>• MIXED – ISO G1, G2, AND V1.19<br>• ISO6BG1 – ISO6B G1<br>• ISO6BG2 – ISO6B G2<br>• ISO6C – The ISO equivalent to EPCglobal UHF Gen 2<br>• ICODE119 – Phillips v1.19 tags (ISO6B emulating EPC tag IDs)<br>• EPCC1G2 – EPCglobal UHF Gen2 |

# PML Adapter

The PML Adapter is used to communicate with the software utility PML core reader. The PMLReader typically listens for connection on one or multiple ports. The adapter properties are shown in the following table.

**TABLE A-5** Adapter Properties for the PMLReader

| Name | Description | Values |
|---|---|---|
| Name | Unique name identifying this adapter. | Sample value = PMLReader |
| classname | Java class name. | com.sun.autoid.adapter.pml.PMLAdapter |
| **Common Properties** | See TABLE 2-4 for general description and values | |
| LogLevel | | |
| hostname | IP address or host name of the reader where the PMLReader is executing. | |
| port | | Must match the corresponding entry in the Simulator.properties of the PMLReader. |

**TABLE A-5** Adapter Properties for the PMLReader *(Continued)*

| Name | Description | Values |
|---|---|---|
| `readerepc` | | Must match the corresponding entry in the `Simulator.properties` of the PMLReader. |
| `autoread` | PMLReader supports autoread mode. | Setting this property to true causes the PMLReader to begin generating simulated RFID events. |
| `scanDuration` | | |

# Adapter for Alien Readers

This adapter communicates with the Alien ALR-9780 or Alien NanoScanner 915 RFID readers. The adapter properties are shown in the following table.

**TABLE A-6** Adapter Properties for the Alien Reader

| Property Name | Description | Values |
|---|---|---|
| `Name` | Unique name identifying this adapter. | Sample value = `AlienReader` |
| `classname` | Java class name. | `com.sun.autoid.adapter.alien.NanoScannerAdapter` |
| **Common Properties** | See TABLE 2-4 for general description and values | |
| `LogLevel` | | |
| `hostname` | | Must be configured with a static IP address, or to boot from DHCP and obtain the IP address specified in this property. |
| `port` | | Default = 23 |
| `readerepc`<br>`autoread`<br>`communicationTimeout`<br>`scanDuration` | | |

**TABLE A-6** Adapter Properties for the Alien Reader *(Continued)*

| Property Name | Description | Values |
|---|---|---|
| **Additional Properties** | Also see "Additional Properties Information for Alien Reader" on page 149. | |
| readerepcANT0, readerepcANT1, and so on. | The EPC identifier associated with antenna 0, 1, 2 or 3 of the reader. | |
| antennaSequence | Specifies the order in which antennae are read. This property must be set to enable the various antennae. | A comma-separated list of antenna identifiers: 0, 1, 2, 3, that depends on the number of antennae supported by the reader.<br>An example value = 0,1 – The reader would pulse antenna 0, followed by antenna 1. Also see the Alien Reader documentation. |
| username | Specifies the user name that the RFID Event Manager uses to communicate with the reader. | Sample value = alien |
| password | Specifies the password that the RFID Event Manager uses to communicate with the Alien reader. | Sample value = password |
| persisttime | Specifies the number of seconds the reader should persist the tag list in its internal tag list buffer. | Sample value = 2, which tells the reader to remember tags for up to 2 seconds before it stops reporting them. |
| AcquireMode | Specifies the mode to be used for collection of tags. Refer to the Alien Reader documentation. | Values are:<br>• Inventory<br>• GlobalScroll<br>Inventory mode is recommended for applications that expect multiple tags to be detected by the reader at once. |

## Additional Properties Information for Alien Reader

- readerepcANT0, readerepcANT1 – This property overrides the value specified by the readerepc property for this particular antenna. (The readerepc property applies to the entire system as long as an EPC has not been specified for a particular antenna.) This property enables the events from two or more antennae from a single reader to be handled by two or more separate processing chains of filters and connectors. Setting the EPC identifier for the antenna tells the RFID Event Manager to discriminate between the antennae if they are enabled, but enabling the antenna is done through the use of the antennaSequence property.
- persisttime – This property should not be confused with the functionality provided by the Smoothing filter. The persisttime property is managed by the Alien RFID reader, not the RFID Event Manager. Refer to the Alien Reader documentation.

# Adapter for ThingMagic Mercury 3 and Sensormatic SensorID Agile 1 Readers

This adapter communicates with the ThingMagic Mercury 3 and the Sensormatic SensorID Agile 1 RFID Reader. The adapter properties are shown in the following table.

**TABLE A-7** Adapter Properties for the Mercury 3 and Agile 1 Readers

| Name | Description | Values |
|---|---|---|
| Name | Unique name identifying this adapter. | Sample value = ThingMagicReader |
| classname | Java class name. | com.sun.autoid.adapter.tyco.Mercury3Adapter |
| **Common Properties** | See TABLE 2-4 for general description and values | |
| LogLevel | | |
| hostname | | Must be configured with a static IP address, or to boot from DHCP and obtain the IP address specified in this property. |
| port | | default = port 80 |

**TABLE A-7** Adapter Properties for the Mercury 3 and Agile 1 Readers *(Continued)*

| Name | Description | Values |
|---|---|---|
| readerepc<br>autoread<br>communicationTimeout<br>scanDuration | | |
| **Additional Properties** | Also see "Additional Properties Information for Mercury 3 and Agile 1 Reader" on page 150. | |
| protocol | Use this property to specify the protocol to be used during data capture. At this time, only one protocol can be active at a time. | This reader supports two RF protocols, CC915 for UHF, and CC1356 for HF.<br>The default value = CC915. |
| readerepcUHF1,<br>readerepcUHF2,<br>and so on. | readerepcUHF1 specifies the EPC for antenna UHF1. This is the EPC reader value associated with the UHF1, UHF2, HF1, and HF2 antennae of the reader. Replace UHF1 with UHF2, HF1, or HF2 for the other antennae. | A sample value = urn:epc:tag:gid-96:1.2.1<br>See "Additional Properties Information for Mercury 3 and Agile 1 Reader" on page 150. |
| antenna | This reader supports two antennae for each protocol: UHF1, UHF2 and HF1, HF2.<br>This property specifies the antenna from which to read. | The default is to enable all the antennae for the RF protocol that has been enabled (see protocol). For example, if CC915 (the default) is selected, both antennae UHF1 and UHF2 will be active.<br>See "Additional Properties Information for Mercury 3 and Agile 1 Reader" on page 150 |

## Additional Properties Information for Mercury 3 and Agile 1 Reader

- readerepc*UHF1* – The variable portion of the property name, *UHF1*, can be UHF1, UHF2, HF1, or HF2. This is the EPC reader value associated with the UHF1, UHF2, HF1, and HF2 antennae of the reader. This property overrides the value specified by readerepc for this particular antenna. The readerepc property applies to the entire system as long as an EPC has not been specified for a particular antenna. This property is useful for handling the events from two or more antennae from a single reader by two or more separate processing chains of filters and connectors.

**Note –** It is not necessary to set the `antenna` property to enable the various antennae. Setting the EPC identifier for the antenna instructs the RFID Event Manager to discriminate between the antennae if they are enabled. Enabling a specific antenna is done through the use of the `antenna` property.

To activate only UHF1, use the following `antenna` property:

```
        <ems:properties>
            <ems:property>antenna</ems:property>
            <ems:value>1</ems:value>
        </ems:properties>
```

**Note –** Programming (writing to) an RFID tag can only be done from antenna 1 (UHF1).

## Adapter for ThingMagic Mercury4 and Sensormatic Agile 2 Readers

This adapter is used to communicate with the ThingMagic Mercury4 and the Sensormatic Agile 2 RFID Reader. The adapter properties are shown in the following table.

**TABLE A-8** Adapter Properties for Mercury4 and Agile 2 Readers

| Property Name | Description | Values |
|---|---|---|
| Name | Unique name identifying this adapter. | Sample value = `ThingMagicReader` |
| `classname` | Java class name. | `com.sun.autoid.adapter.tyco.Mercury4Adapter` |
| **Common Properties** | See TABLE 2-4 for general description and values | |
| `LogLevel` | | |
| `hostname` | | Must be configured with a static IP address, or to boot from DHCP and obtain the IP address specified in this property. |

**TABLE A-8** Adapter Properties for Mercury4 and Agile 2 Readers *(Continued)*

| Property Name | Description | Values |
|---|---|---|
| port | | default = 80 |
| readerepc<br>autoread<br>communicationTimeo ut<br>scanDuration | | |
| **Additional Properties** | Also see "Additional Properties Information for Mercury 3 and Agile 1 Reader" on page 150. | |
| protocol | Use this property to specify the protocol to be used during data capture. | This reader supports three data capture protocols:<br>• EPC0<br>• EPC1<br>• ISO18000-6B<br>Default = ALL |
| readerepcUHF1, readerepcUHF2,and so on. | readerepcUHF1 specifies the EPC for Antenna UHF1.<br>This EPC reader value is associated with the corresponding UHF1, UHF2, UHF3, UHF4, UHF5, UHF6, UHF7 and UHF8 antennae.<br>Replace UHF1 with UHF2, UHF3 and so on for the other antennae. | A sample value = urn:epc:tag:gid-96:1.2.1. |
| antenna | This reader supports two antennae for each protocol: UHF1, UHF2 and HF1, HF2. This property specifies the antenna from which to read. | The default is to enable all the antennae for the RF protocol that has been enabled (see protocol). For example, if CC915 (the default) is selected, both antennae UHF1 and UHF2 will be active. |

# Adapter for Matrics Readers

This adapter is used to communicate with the Matrics RDR-001 and Matrics AR-400 RFID readers. The adapter properties are shown in the following table.

**Note –** The use of a serial to network (Ethernet) adapter is required.

**TABLE A-9** Adapter Properties for the Matrics Reader

| Property Name | Description | Values |
|---|---|---|
| Name | Unique name identifying this adapter. | Sample name = MatricsReader |
| classname | Java class name. | com.sun.autoid.adapter.matrics.MatricsReaderAdapter |
| **Common Properties** | See TABLE 2-4 for general description and values | |
| LogLevel | | |
| hostname | IP address for the network-facing interface of the Serial to Ethernet Convertor Port which interfaces to the Matrics RDR-001. Future Matrics readers will have this built in. | example value = 192.168.2.150 |
| port | IP port for the network- facing interface of the Serial to Ethernet Convertor Port which interfaces to the Matrics RDR-001. Future Matrics readers will have this built in. | |
| readerepc | | |
| autoread | | |
| communicationTimeout | | |
| scanDuration | Not used by this reader | |
| **Additional Properties** | Also see "Additional Properties Information for Matrics Reader" on page 154. | |
| ConnectionType | Specifies the type of network being used to connect to the reader. | network – Indicates IP connection<br>serial – Not currently used |
| readerepcANTx | Specifies an EPC value associated with one of the specific multiple antennae on the reader, where the variable *x* is an EPC value associated with antenna 1, 2, 3, and 4 of the reader. | Sample value = urn:epc:tag:gid-96:1.1.1 |
| Node | See "Additional Properties Information for Matrics Reader" on page 154 | |

**TABLE A-9** Adapter Properties for the Matrics Reader *(Continued)*

| Property Name | Description | Values |
|---|---|---|
| Antenna*n* | Specifies whether a particular antenna is to be used where *n*, indicates the number of the antenna. This readers supports antenna 1-4. | A value of 1 means it is to be used. A value of zero means it is to be inhibited. For example, to enable Antenna 1, the following is used:<br>`<ems:properties>`<br>`<ems:property>Antenna1</ems:property>`<br>`<ems:value>1</ems:value>`<br>`</ems:properties>` |
| Power | Controls the power level transmitted during a given read or write interval | The minimum value is `01hex`. The maximum value is `FFhex,` which specifies a full power condition (in this case 4 watts). The power level is logarithmic. For the RDR-001 reader, the maximum power is about `30dBm`. The value `C0hex` is about 50 percent and `80hex` about 25 percent of the maximum power. |
| Environment | See | |
| CombinedAntenna | | |
| FilterLength | Not currently used. | |
| Filter | Not currently used. | |
| debugflags | Not currently used. | |

## Additional Properties Information for Matrics Reader

- `readerepcANT`*n* – Specifies the reader antenna EPC identifier, an EPC value associated with antenna 1, 2, 3, and 4 of the reader where the variable *n* is the number of the antenna. This property overrides the value specified by `readerepc` for this particular antenna. The `readerepc` property applies to the entire system as long as an EPC has not been specified for a particular antenna. This property is useful for handling the events from two or more antennae from a single reader by two or more separate processing chains (filters and connectors). See `readerepc`. The following example specifies the EPC for antenna 1. Replace 0 with 1, 2, or 3 for the other antennae.

```
<ems:properties>
    <ems:property>readerepcANT0</ems:property>
    <ems:value>urn:epc:tag:gid-96:1.1.1</ems:value>
</ems:properties>
```

`readerepcANT0` corresponds to physical antenna 1, `readerepcANT1` corresponds to antenna 2, `readerepcANT2` corresponds to antenna 3, and `readerepcANT3` corresponds to antenna 4.

---

**Note –** It is necessary to set the `antenna`*n* (where *n* is the number of the antenna) property to 1 to enable the use of antenna 1. Setting the EPC identifier for antenna 1 tells the RFID Event Manager to discriminate between the antennae if they are enabled, but enabling the antenna is done through the use of the `antenna`*n* property.

---

- `Node` – The Matrics reader sits on an RS 485 bus, a form of serial bus where multiple serial devices can operate, unlike RS 232, where only one device can operate at a time. Each node on a RS 485 bus must have an address. The serial-to-network adapter relays commands to the reader based on this address. If additional readers are present then they have other addresses. For normal operation, no more than one reader is on the RS 485 bus. Each reader has its own serial-to-network adapter port interface and is directly IP-addressable. The following RS 485 address must be the same for all readers.

```
<ems:properties>
    <ems:property>Node</ems:property>
    <ems:value>4</ems:value>
</ems:properties>
```

- `Power` – This property controls the power level transmitted during a given read or write interval. The normal transmit power of a North American Matrics RDR-001 reader is 4 watts. In other locales, this value might be less. The actual value is a percentage value. The minimum value is `01hex`. The maximum value is `FFhex`, which specifies a full power condition (in our case 4 watts). The power level is logarithmic. With the reader RDR-001, the maximum power is about 30dBm. The value `C0hex` is about 50 percent and `80hex` is about 25 percent of the maximum power.

```
<ems:properties>
    <ems:property>Power</ems:property>
    <ems:value>C0</ems:value>
</ems:properties>
```

- `Environment` – This property determines how long the reader tries to read tags during a Read Full command. A larger number means longer, more intense reading (more frequencies in an FCC part 15 reader are used). This property is useful for applications where tagged items are not moving (stationary), such as in a shelf application, to overcome issues with interference and RF-Null’s on a fixed pool of tags.

In an environment where tags move in and out of the read field of a reader, it is important to read as fast as possible to be able to start negotiating with new tags coming into the read field. That is why in dynamic environments the variable is usually small. The environment property is dependent on the location of the system. Generally, it is safe to start with the smallest value (00 for dynamic, 04 for static) to evaluate the performance. To improve reads, adjusting the property up or down might be necessary.

Practically, this value should remain at 4. There are few benefits to changing this value, unless you are sure that your environmental conditions are relatively constant.

```
<ems:properties>
    <ems:property>Environment</ems:property>
    <ems:value>4</ems:value>
</ems:properties>
```

- CombinedAntenna – This property indicates that all the antennas marked with “included” are grouped. That means a host has only to address the antenna with the smallest index to get reads from all combined antennas. For example, if antenna 1, 3 and 4 are combined, the host executes Read Full *only* for antenna 1 to get reads from antenna 1, 3 and 4.

  This property only works for antenna 1. The reader tends to produce antenna faults whenever a value other than antenna 1 is chosen.

  Because you generally want to distinguish between antennas, using this property is discouraged and is not connected. The property has been kept in case those interested in using it want to make future modifications.

```
<ems:properties>
    <ems:property>CombinedAntenna</ems:property>
    <ems:value>1</ems:value>
</ems:properties>
```

- Wait – Specifies the reader wait time that is used by the underlying adapter framework to pause between read cycles. The normal operation of the reader is to issue a read command, because autoread is simulated, one read command is required for each read cycle. The reader returns all tags read during this time, after which, the adapter framework waits for the time specified by this property.

```
<ems:properties>
    <ems:property>Wait</ems:property>
    <ems:value>80</ems:value>
</ems:properties>
```

# Adapter for Printronix RFID Printers

This adapter is used to communicate with the Printronix SL 5000e RFID printer. The adapter properties are shown in following table.

**TABLE A-10** Adapter Properties for the Printronix SL 5000e RFID Printer

| Name | Description | Values |
|---|---|---|
| `Name` | Unique name identifying this adapter. | Sample name = `PrintronixPrinter` |
| `classname` | Java class name. | `com.sun.autoid.adapter.printronix.PrintronixAdapter` |
| **Common Properties** | See TABLE 2-4 for general description and values | |
| `LogLevel`<br>`hostname`<br>`port`<br>`readerepc`<br>`autoread`<br>`communicationTimeout` | | |
| `scanDuration` | not used by this device | |
| **Additional Properties** | . | |
| `template` | Specifies the location of the default print template. | `/tmp/templatefile.txt` |
| template.*variable* | Template identified by word `variable`. There can be zero or more occurences of this property, each with a unique substitution for *variable*. | The value can be either a URL or local file.<br>For example, if name =`template.case`, then sample value = `/tmp/templatecase.txt`. |

# Adapter for SAMSys Readers

This adapter is used to communicate with the SAMSys MP9320 EPC V2.7 Reader. The adapter properties are shown in the following table.

**TABLE A-11** Adapter Properties for the SAMSys Reader

| Name | Description | Values |
|---|---|---|
| Name | Unique name identifying this adapter. | Sample value = SAMSysReader |
| classname | Java class name. | com.sun.autoid.adapter.SAMSys.SAMsysAdapter |
| **Common Properties** | See TABLE 2-4 for general description and values | |
| LogLevel | | |
| hostname | IP address or host name of the reader where the SAMSys reader is executing | Must be configured with a static IP address, or to boot from DHCP and obtain the IP address specified in this property. |
| port | | Default = 2001 |
| readerepc<br>autoread | | |
| scanDuration | See "Additional Properties Information for SAMSys Reader" on page 161. | |
| communicationTimeout | See "Additional Properties Information for SAMSys Reader" on page 161. | |
| **Additional Properties** | | |
| enableCommandResponseCRC | Command Response Cyclic Redundancy Check (CRC). See also "Additional Properties Information for SAMSys Reader" on page 161. | true – Perform check<br>false – Omit check |
| enableTagReadResponseCRC | Response Cyclic Redundancy Check (CRC). See also "Additional Properties Information for SAMSys Reader" on page 161. | true – Perform check<br>false – Omit check |

**TABLE A-11** Adapter Properties for the SAMSys Reader *(Continued)*

| Name | Description | Values |
|---|---|---|
| protocols | Specifies the type of RF protocol to be scanned for by the reader. When no protocols property is defined, the settings on the reader are used without modification. The reader can scan for multiple types simultaneously. | Multiple protocols can be specified by using a comma-separated list such as, [IS186B,EPC1].<br>Tested values are IS186B and EPC1.<br>See also "Additional Properties Information for SAMSys Reader" on page 161. |
| readerepcANT*n* | EPC value associated with the ANT1, ANT2, ANT3, and ANT4 antennae of the reader, where the variable *n* equals the number of the antenna. | Example of usage:<br>property name = readerepcANT1 and<br>value = urn:epc:tag:gid-96:1.2.11 |
| antennaSequence | Specifies the specific antenna from which to read. The order of the antennas listed in the property determines the jorder in which the antennae are read. | Default is ANT1.<br>To activate a sequence of ANT1 then ANT3 then ANT2, use the following value for antennaSequence – ANT1,ANT3,ANT2 |
| cycles | Specifies the Antenna Inventory Round Operations, which is the number of inventory operations performed on each antenna before moving to the next antenna in the antenna sequence.<br>See also "Additional Properties Information for SAMSys Reader" on page 161. | Default value = 1 |

**TABLE A-11** Adapter Properties for the SAMSys Reader *(Continued)*

| Name | Description | Values |
|---|---|---|
| antenna*n*Power | Specifies the preferred power level, where the variable *n* designates the antenna number. | Values can be entered as hexadecimal or decimal digits. Hexadecimal values must be prefixed with the characters 0x.<br>The following example sets the same power levels for both antenna 1 and antenna 2.<br>• property name = antenna1Power with value = 0x60<br>• property name = antenna2Power with value = 96 |
| buzzerMode | The SAMSys reader produces an audible beep when tags are observed in its field of view. The buzzer is enabled or disabled with this property. | Values can be ON or OFF |
| regulatoryParty | Sets the Transmit Power Configuration (TPC) register within the SAMSys reader.<br>See also "Additional Properties Information for SAMSys Reader" on page 161. | Values are ETSI and FCC |
| observationSetTimeout | Sets the timeout threshhold used to detect the end of a reported tag observation set. This is separate and disctinct from the communicationTimeout property, which is used for all other communication timeouts, except the detection of the end of an observation set. | default 100 msec |

# Additional Properties Information for SAMSys Reader

- communicationTimeout – This property is used exclusively for all commands and responses on the device, except tag read commands and the respective responses. The property defines the inactivity threshold for reader responses. Performance can be highly dependent on this property, therefore reduce it from the default value of 10000 msec (10 seconds). You can start with a value of 1500msec, and make further reductions after observing deployment-specific performance. Observe the read performance with log levels set to FINEST. For more information, refer to TABLE 2-4.

  Both the communicationTimeout and observationSetTimeout properties are used for detecting responses to the read command. The communicationTimeout property is used for the initial response, while observationSetTimeout is used for all subsequent observation responses. communicationTimeout behaves this way for all other commands.

  The observationSetTimeout property sets the inactivity threshold for signalling that an observation set has completed. Default is 100 msec. This value can be reduced after observing deployment-specific performance.

- enableCommandResponseCRC – Specifies the Command Response Cyclic Redundancy Check (CRC). Each command response received from the reader is provided with a CRC checksum. The SAMSysAdapter enables you to verify that the command response arrived without any transmission error. Alternatively, this check can be omitted. This property is used to establish whether these checks are performed. The valid values are true and false.
- enableTagReadResponseCRC – Specifies the Response Cyclic Redundancy Check (CRC). Each reported tag observation sent from the reader is provided with a CRC checksum. The SAMSysAdapter enables you to verify that the transmitted tag data arrived without any transmission error. Alternatively, this check can be omitted to achieve greater system throughput. This property is used to establish whether these checks are performed. The valid values are true and false.
- scanDuration – The SAMsys reader does not offer the ability to poll for a specified duration as some other reader devices do. The adapter for the SAMSys reader emulates this capability. The adapter waits for the end of transmission from the reader (which could exceed the specified value) and dwells for the remaining time if the elapsed time is less than the value specified in the scanDuration property.
- protocol – The SAMSys reader supports reading a number of different RFID tags. The value of this property specifies the types being used during data capture. The default is ALL. Multiple property definitions can be combined by using a comma-separated list.The following protocols have been tested:
  - For ISO 18000-6A, the value is IS186A.

  - For ISO 18000-6B, Intermec Intellitag, Philips I-CODE HSL, the value is `IS186B`.
  - For Alien EPC, the value is `EPC1`.
  - For EM 4022, EM4222 the value is `STG`.
  - For EPC0, the value is `EPC0`.
- `readerepcANT`*n* – Specifies the EPC value associated with the ANT1, ANT2, ANT3,  and ANT4  antennae of the reader, where the variable *n* equals the number of the antenna. This property overrides the value specified by `readerepc` for this particular antenna. The `readerepc` property applies to the entire system as long as an EPC has not been specified for a particular antenna. The `readerepcANT`*n* property is useful for specifying that the events from two or more antennae from a single reader be handled by two or more separate processing chains. See also `readerepc`.

---

**Note –** It is not necessary to set the `antenna` property to enable the use of the various antennae. Setting the EPC identifier for the antenna instructs the RFID Event Manager to discriminate between the antennae if they are enabled. Enabling a specific antenna is done through the use of the `antennaSequence` property.

---

- `cycles` – Specifies the Antenna Inventory Round Operations. The SAMSys reader enables the Multiplexer Configuration Word (MCW) register to be set to specify the number of inventory operations per round. This property defines the number of inventory operations performed on each antenna before moving to the next antenna in the antenna sequence. See the SAMSys Comprehensive Heuristic Unified Messaging Protocol in the SAMSys documentation for more detailed information.
- `antenna`*n*`Power` – Specifies the desired power level. The SAMSys reader has 48 power settings. The lowest setting is 60, translating to 12 dBm and 16 mW. 02 corresponds to 34.7 dBm 2.95 W.

```
60, 5e, 5c, 5a, 58, 56, 54, 52, 50,
4e, 4c, 4a, 48, 46, 44, 42, 40,
4e, 4c, 4a, 48, 46, 44, 42, 40,
3e, 3c, 3a, 38, 36, 34, 32, 30,
2e, 2c, 2a, 28, 26, 24, 22, 20,
1e, 1c, 1a, 18, 16, 14, 12, 10,
0e, 0c, oa, 08, 06, 04, 02
```

- `regulatoryParty` – Sets the Transmit Power Configuration (TPC) register within the SAMSys reader. Setting the value of this property to `FCC` sets the reader for FCC operation. Setting the value to `ETSI` sets the reader to ETSI operation with frequency hopping. Leaving the property unset leaves the reader register unchanged. SAMSys recommends changing this parameter only at the direction of SAMSys personnel. See the SAMSys Comprehensive Heuristic Unified Messaging Protocol in the SAMSys documentation for more detailed information.

## Adapter for Symbol MC9000-G Readers

This adapter communicates with a custom software program hosted on the Symbol MC9000 device. This adapter opens a server socket over TCP and listens for connections from the Symbol MC9000-G device. The data payload consists of one or more lines of text with two possible formats containing the data fields as shown in the following table.

**TABLE A-12** Adapter Data Fields and Descriptions for the Symbol MC9000-G Reader

| Data Field | Description | Example |
|---|---|---|
| `tag` | Hexadecimal representation of an EPC starting with the letter H. | `H40001403EA000001` |
| `count` | An integer representing the number of times the EPC was read by the device. | `1` |
| `timestamp` | The format is "yyyy-MM-dd'T'HH:mm:ss.SSSZ", where Z is the OffsetTimeZone: Sign Hours::Minutes. | `2005-07-15T11:23:10-05:00` |
| `classType` | The letter `C` followed by the class type. | C1 |

- **Format A** – "tag,count\n". An example of Format A looks similar to the following:
  - `H40001403EA000001,1`
  - `H40001403EA000002,3`
  - `H40001403EA000028,15`

  In this example, the adapter receives three 64-bit EPC tags from the reader. The first tag was detected once, the second tag was detected three times, and the third tag was detected fifteen times. In release 3.0, the adapter ignores the count, and does not pass it to the event listeners.
- **Format B** – "timestamp,tag,classType,count\n". An example of Format B looks similar to the following:
  - `2005-07-15T11:23:10-05:00,H40001403EA000001,C1,1`
  - `2005-07-15T11:23:13-05:00,H40001403EA000002,C1,3`
  - `2005-07-15T11:23:20-05:00,H40001403EA000028,C1,15`

> In this example, the adapter receives three class 1, 64-bit EPC tags from the reader. The first tag was detected once, the second tag was detected three times, and the third tag was detected fifteen times. In release 3.0 , the time that the tag was read is used as the current time. The adapter ignores the count and class type, which are not passed up stream to the event listeners.

Any application hosted on the Symbol MC9000-G device that conforms to this protocol will suffice. In most cases, these applications are built for a specific engagement. Contact your Sun Microsystems, Inc. representative to obtain a sample hand-held application that reads the contents of the comma-separated `RFIDTagList.csv` file that is generated by the Symbol RFID sample application that is shipped with your device. After reading this file, the sample Sun application sends the tag data that is read to this adapter using the format previously defined.

Applications are expected to use the RFID Software reader or web service APIs to query the tag list. In this case, set the `autoread` property to false, and have the application control how often to query for tags. Alternatively, a connector can be chained to this adapter to post a list of tags to a listening application. In this case, you might set the `autoread` property to true. Setting the `autoread` property to true enables a periodic set of queries can be generated to the handheld device and the list of tags posted when the reader posts its data.

**Note –** The list of tags is only valid for a single query or posting.

Two consecutive queries to the adapter for a list of tags results in two separate attempts to collect RFID data from the device. Typically, the adapter queries the device for a list of tags on cycle `n`, resulting in tag list `L1`. On attempt `n+1`, the adapter might get an empty list if the hand-held device is not connected. On attempt `n+2`, the adapter might get tag list `L2`. A tag list is erased from the adapter buffer after the list is posted to its listeners.

The adapter properties for this reader are shown in TABLE A-13.

**TABLE A-13** Adapter Properties for the Symbol MC9000-G Reader

| Name | Description | Values |
|---|---|---|
| `Name` | Unique name identifying this adapter. | Sample value = `SSocketAdapter` |
| `classname` | Java class name. | `com.sun.autoid.adapter.ssocket.SSocketAdapter` |
| **Common Properties** | See TABLE 2-4 for general description and values | |
| `LogLevel`<br>`hostname`<br>`readerepc`<br>`autoread` | | |

**TABLE A-13** Adapter Properties for the Symbol MC9000-G Reader *(Continued)*

| Name | Description | Values |
|---|---|---|
| port | Specifies the port on which to create a server socket. | Default = 59000 |
| scanDuration | | Default = 15,000 msecs |
| datafmt | This property indicates the data format. Currently, this adapter supports only the comma-separated values (CSV) format described above. | |

# Adapter for Zebra Technologies Printers

This adapter is used to communicate with the Zebra Technologies R110XiIIIPlus Printer. The adapter properties are shown in the following table.

**TABLE A-14** Adapter Properties for Zebra Technologies RFID Printer

| Name | Description | Values |
|---|---|---|
| Name | Unique name identifying this adapter. | Sample value = ZebraPrinter |
| classname | Jjava class name. | com.sun.autoid.adapter.zebra.ZebraR110XiAdapter |
| **Common Properties** | See TABLE 2-4 for general description and values | |
| LogLevel<br>hostname<br>port<br>readerepc<br>autoread<br>communicationTimeout | | |
| scanDuration | Not used by this device | |

**TABLE A-14** Adapter Properties for Zebra Technologies RFID Printer *(Continued)*

| Name | Description | Values |
|---|---|---|
| **Additional Properties** | . | |
| `template` | Specifies the location of the default print template. | `/tmp/templatefile.txt` |
| template.*variable* | Template identified by the *variable*. There can be zero or more occurences of this property, each with a unique substitution for *variable*. | The value can be either a URL or a local file.<br>For example, if the property name = `template.case`, then the value = `/tmp/templatecase.txt` . |
| `description` | Specifies a description to be printed on the label. | Value = label description text |

APPENDIX **B**

# RFID Event Manager Component Reference

This appendix describes the properties of the RFID Event Manager components included with the RFID Software. The following tables list the properties with a description and the valid values. Refer to the property descriptions and values to determine the specific settings for your RFID system. The components are used to create RFID Event Manager roles as described in Chapter 2. See "To Define RFID Event Manager Roles" on page 30.

# Smoothing Filter

This component creates a union of EPCs that are discovered over the number of specified *n* cycles. If an EPC was discovered in cycle < *n*, it is reported. If an EPC has not been viewed in more than the last *n* cycles, it is not reported. This component is necessary because the RFID readers do not report tags with 100% accuracy.

**TABLE B-1** Smoothing Filter Properties

| Property | Description | Value |
|---|---|---|
| `classname` | Java class name. | `com.sun.autoid.filter.Smoothing` |
| `LogLevel` | Defines a set of standard logging levels that can be used to control logging output. | Possible values are:<br>• SEVERE (highest value)<br>• WARNING<br>• INFO<br>• CONFIG<br>• FINE<br>• FINER<br>• FINEST (lowest value) |
| `MaxCycles` | The maximum read cycles that an item stays in the virtual view of the reader. | |
| `MaxPersistTime` | The maximum time (in msec) to wait for a reader message before issuing a prune event. | |

# Delta Filter

This component reports RFID tags leaving and entering the radio frequency fields. For example, on a reader, if at time T1, two EPCs were discovered, then two TagsIn events are reported. If at time T2, one of the EPCs disappeared, then a TagsOut event is reported.

**TABLE B-2** Delta Filter Properties

| Property | Description | Value |
|---|---|---|
| `classname` | Java class name. | `com.sun.autoid.filter.Delta` |
| `LogLevel` | Defines a set of standard logging levels that can be used to control logging output. | Possible values are:<br>• SEVERE (highest value)<br>• WARNING<br>• INFO<br>• CONFIG<br>• FINE<br>• FINER<br>• FINEST (lowest value) |
| `reportIn` | Specifies whether to report the incoming tags. | Default = true |
| `reportOut` | Specifies whether to report the leaving tags. | Default = true |
| `reportOnEmpty` | Specifies whether to report even if no tags have changed. | Default = false |
| `useRecord` | Specifies whether to use recorded events from the previous run. | Default = false |
| `RecordRoot` | Specifies the path to get and save the events. | `Default = java.io.tmpdir` |

# EPC Filter

This component performs a *pass* filter on the value of the RFID tag EPC values. Those EPC values that match the specified EPC pattern are passed on to listeners, while others are not.

**TABLE B-3** EPC Filter Properties

| Property Name | Description | Example |
|---|---|---|
| `classname` | Java class | `com.sun.autoid.filter.EPCFilter` |
| `LogLevel` | Defines a set of standard logging levels that can be used to control logging output. | Possible values in addition to ALL are:<br>• SEVERE (highest value) - fewest messages<br>• WARNING<br>• INFO<br>• CONFIG<br>• FINE<br>• FINER<br>• FINEST (lowest value) most messages |
| `Mask` | An identity URI pattern used for matching. | Examples are:<br>• For a 96-bit GID tag mask – `urn:epc:pat:gid-96:10.1002.[1-20]`<br>• For a 64-bit SGTIN tag mask – `urn:epc:pat:sgtin-64:7.1234567.100734.[1-20]`<br>• For a 64-bit giai tag mask – `urn:epc:pat:giai-64:7.0614141.[2-4]` |
| `EPC` | EPC identifier URI for the matching id. | Examples are:<br>• For a 96-bit GID EPC – `urn:epc:tag:gid-96:10.1002.50`<br>• For a 64-bit SGTIN EPC – `urn:epc:tag:sgtin-64:7.1234567.100734.20`<br>• For a 64-bit giai EPC – `urn:epc:tag:giai-64:7.0614141.4` |

# BandPass Filter

The BandPass filter is used to select a subset of events that match the specified event source's identity. If an event matches, the Event object is passed to the listeners of the filter, otherwise the Event object is dropped.

An event source identity matches if any of the following conditions are true:

- An EPC identifier property has been specified in the configuration for this filter and the event source is an identifier that matches exactly the specified EPC identifier.
- An EPC String has been specified in the configuration for this filter and the event source is a String that matches exactly the specified EPC String
- A Mask property has been specified in the configuration for this filter and the event source is an identifier that matches the specified Mask's identity URI pattern.

**TABLE B-4** BandPass Filter Properties

| **Property Name** | **Description** |
|---|---|
| classname | com.sun.autoid.filter.BandPass |
| LogLevel | Defines a set of standard logging levels that can be used to control logging output. Possible values are:<br>• SEVERE (highest value)<br>• WARNING<br>• INFO<br>• CONFIG<br>• FINE<br>• FINER<br>• FINEST (lowest value) |
| Mask | An identity URI pattern used for matching. See TABLE B-3 for examples. |
| EPC | An EPC identifier URI or a String for the matching ID. If EPC is an identifier URI, then the match is successful only if the event's source is an identifires that matches the value of this EPC property. If the EPC property is not an identifier URI, then it is treated as a String and the event's source must be a String that matches the EPC value |

# FileLogger Connector

This component provides a general connector for the RFID Event Manager and writes PML core to an output file.

**TABLE B-5** FileLogger Properties

| Property Name | Description |
|---|---|
| `classname` | `com.sun.autoid.logger.FileLogger` |
| `LogLevel` | Defines a set of standard logging levels that can be used to control logging output. Possible values are:<br>• SEVERE (highest value)<br>• WARNING<br>• INFO<br>• CONFIG<br>• FINE<br>• FINER<br>• FINEST (lowest value) |
| `File` | Destination file where the output is written. |
| `Append` | Specifies whether to append to the file. Append to the file if true. Starts a new file at the beginning if false. |

# HttpPMLLogger Connector

This component provides a connector that writes PML core to an HTTP connection.

**TABLE B-6** HttpPMLLogger Properties

| Property | Description |
|---|---|
| `classname` | com.sun.autoid.logger.HttpPMLLogger |
| `LogLevel` | Defines a set of standard logging levels that can be used to control logging output. Possible values are:<br>• SEVERE (highest value)<br>• WARNING<br>• INFO<br>• CONFIG<br>• FINE<br>• FINER<br>• FINEST (lowest value) |
| `URL` | Destination HTTP address to connect to. |
| `Proxy` | Optional proxy host. |
| `ProxyPort` | Optional proxy port, default is 80 if proxy host is set. |

# JMSLogger Connector

This component provides a connector that sends PML core events using a JMS Message.

**TABLE B-7** JMSLogger Properties

| Property | Description |
|---|---|
| `classname` | com.sun.autoid.logger.JMSLogger |
| `LogLevel` | Defines a set of standard logging levels that can be used to control logging output. Possible values are:<br>• SEVERE (highest value)<br>• WARNING<br>• INFO<br>• CONFIG<br>• FINE<br>• FINER<br>• FINEST (lowest value) |
| `JndiContextFactory` | Sets the `java.naming.factory.initial` property with the default value of `com.sun.jndi.fscontext.RefFSContextFactory`. |
| `JndiProviderURL` | `java.naming.provider.url.` |
| `Principal` | `java.naming.security.principal` |
| `Credentials` | `java.naming.security.credentials` |
| `Authentication` | `java.naming.security.authentication` |
| `ConnectionFactory` | connection factory |
| `UserName` | optional user name for creating connections |
| `UserPassword` | optional user password for creating connections |
| `QueueName` | queue name |
| `TopicName` | topic name |
| `Transacted` | Indicates whether session is transacted, default false |
| `AcknowledgeMode` | Indicates whether the consumer or the client acknowledges the messages received.<br>Default value = `Session.AUTO_ACKNOWLEDGE` |
| `SendPML` | If true, sends PML as a text message; otherwise sends the event object as the message (default is true). |

# NullConnector Filter

The NullConnector filter consumes events and does nothing with them. The primary purpose of this connector is to terminate processing of the the event chain if no external processing is required.

**TABLE B-8** NullConnector Properties

| Property | Description |
|---|---|
| `classname` | `com.sun.autoid.logger.NullConnector` |
| `LogLevel` | Defines a set of standard logging levels that can be used to control logging output. Possible values in addition to ALL are:<br>• SEVERE (highest value) – fewest messages<br>• WARNING<br>• INFO<br>• CONFIG<br>• FINE<br>• FINER<br>• FINEST (lowest value) most messages |

# SocketLogger Connector

This component creates a socket connection and starts writing PML core to the connection.

**TABLE B-9** SocketLogger Properties

| Property | Description |
|---|---|
| `classname` | com.sun.autoid.logger.SocketLogger |
| `LogLevel` | Defines a set of standard logging levels that can be used to control logging output. Possible values are:<br>• SEVERE (highest value)<br>• WARNING<br>• INFO<br>• CONFIG<br>• FINE<br>• FINER<br>• FINEST (lowest value) |
| `Host` | Target host for the connection. |
| `Port` | Target port for the connection. |
| `CloseOnSend` | If true, closes the socket connection after each message |

# SSocketLogger Connector

This component creates a server socket and, when the socket connection is accepted, starts writing PML to the connection.

**TABLE B-10** SSocketLogger Properties

| Property Name | Description |
|---|---|
| `classname` | com.sun.autoid.logger.SSocketLogger |
| `LogLevel` | Defines a set of standard logging levels that can be used to control logging output. Possible values are:<br>• SEVERE (highest value)<br>• WARNING<br>• INFO<br>• CONFIG<br>• FINE<br>• FINER<br>• FINEST (lowest value) |
| `Port` | Target port on which to listen for connections. |
| `CloseOnSend` | if true, closes the socket connection after each message. |

# SAPAII Connector

See "Architecture Overview of the RFID Software Plug-In for SAP AII" on page 62.

**TABLE B-11** SAPAII ConnectorProperties

| **Property** | **Description\Value** |
|---|---|
| `classname` | `com.sun.autoid.logger.SapLogger` |
| `LogLevel` | Defines a set of standard logging levels that can be used to control logging output. Possible values are:<br>• SEVERE (highest value)<br>• WARNING<br>• INFO<br>• CONFIG<br>• FINE<br>• FINER<br>• FINEST (lowest value) |
| `SapAIIUrl` | The URL where the SAP AII software listens for notifications. |
| `SapDeviceControllerName` | The device controller ID used in notifications to the SAP AII software. |
| `SapCommand` | The default command to include in the notification to SAP AII software. The default value is `IN`. |

# EPCGui Connector

The EPCGui connector is a graphical user interface (GUI) that displays RFID tags and the number of consecutive times the tags have been sensed by a reader. The tags change colors from red to yellow to green to indicate the number of times they have been viewed.

- Red – The tag has just been discovered.
- Yellow – The tag has been in the reader's sensing field for over 10 cycles.
- Green – The tag has been in the reader's sensing field for 30 cycles or more.

This GUI runs inside the same Java virtual machine (Java VM) as the rest of the Execution Agent. To display the GUI on a different system, the shell that launched the Execution Agent must be remotely displayed.

To avoid this limitation, use the `RemoteEventProducer` connector, which posts events to the TagViewer that can run independently from the Execution Agent.

The EPCGui is useful for debugging. Use the combination of a `RemoteEventProducer` and TagViewer in a production environment. The system automatically creates a `RemoteEventProducer` connector for every adapter. This enables the TagViewer to receive nonfiltered tags detected by the reader. If you want to see only filtered tags, then you need to attach a `RemoteEventProducer` connector to the output of the filter. Then use the TagViewer to see the filtered tags.

**TABLE B-12** EPCGui Connector Properties

| Property Name | Description/Value |
|---|---|
| `classname` | `com.sun.autoid.logger.epcui.viewer.GuiLogger` |
| `LogLevel` | Defines a set of standard logging levels that can be used to control logging output. Possible values are:<br>• SEVERE (highest value)<br>• WARNING<br>• INFO<br>• CONFIG<br>• FINE<br>• FINER<br>• FINEST (lowest value) |
| `Title` | A string that appears as the title of the window. |

# REProducer Connector

This component, the remote event producer is a general connector to produce Remote Event objects. This class does not implement guranteed event delivery. If there are no registered consumers of the generated events, the events are lost.

**TABLE B-13** RemoteEventProducer Properties

| Property | Description |
|---|---|
| classname | com.sun.autoid.logger.REProducer |
| LogLevel | Defines a set of standard logging levels that can be used to control logging output. Possible values are:<br>• SEVERE (highest value)<br>• WARNING<br>• INFO<br>• CONFIG<br>• FINE<br>• FINER<br>FINEST (lowest value)<br>Also see the java.util.logging.Level and LogLevel property description in TABLE 2-4. |
| EventID | Integer identifier for the event. The consumer must specify the same EventID in order to discover the events. |
| LogicalReader | Name of the logical reader for this producer. |
| PhysicalReader | Name of the physical reader for this producer. |
| RoundRobin | If true, then use the RoundRobin dispatcher. If false, then dispatch to all registered listeners. |
| MaxIgnoredEvents | Maximum number of times trying to fire an event with no listeners registered before an exception is thrown The default = 25. |

# EpcisJms Connector

The EpcisJms connector sends events from the RFID Event Manager to the RFID Information Server using the Java Message Service (JMS) protocol. This connecter populates the CURRENT_OBSERVATION table and the OBSERVATION_LOG table in the RFID Information Server database. Also see "Configuring the RFID Event Manager to Use an `EpcisJms` Connector" on page 119.

**TABLE B-14** EpcisJms Connector Properties

| Property | Description |
|---|---|
| `classname` | `com.sun.autoid.logger.EpcisLogger` |
| `LogLevel` | Defines a set of standard logging levels that can be used to control logging output. Possible values are:<br>• SEVERE (highest value)<br>• WARNING<br>• INFO<br>• CONFIG<br>• FINE<br>• FINER<br>FINEST (lowest value)<br>Also see `java.util.logging.Level` and `LogLevel` property descriptions in TABLE 2-4. |
| `EpcisUrl` | Specifies the URL for the RFID Information Server. The URL is of the form:<br>`http://`*hostname:portnumber*`/epcis/service` where *hostname* is the name of the machine where you installed your RFID Information Server and the *portnumber* is your HTTP port. |
| `HttpProxyHost` | (Optional) Specifies the host name of the HTTP proxy server. |
| `HttpProxyPort` | (Optional) Specifies the port number for the HTTP proxy server. |
| `UseJms` | (Required) Specifies the delivery method. Value must be one of the following:<br>• `true`<br>• `topic` (see `TopicName` property)<br>• `queue` (see `QueueName` property) |

**TABLE B-14** EpcisJms Connector Properties *(Continued)*

| Property | Description |
|---|---|
| TopicName | (Optional) Only used if the UseJms property is true or topic. Specifies the topic name for JMS.<br>Default value is epcisTopic. |
| QueueName | (Optional) Only used if the UseJms property is queue. Specifies the queue name for JMS.<br>Default value is epcisQueue. |
| ConnectionFactory | (Optional) Only use if the UseJms property is true, topic, or queue. Specifies the Java Naming and Directory Interface (JNDI) name of the connection factory.<br>When UseJms property is true or topic , this property is TopicConnectionFactory<br>When UseJms property is queue, this property is QueueConnectionFactory. |
| java.naming.security.principal | RFID Information Server user name. |
| java.naming.security.credentials | RFID Information Server user password. |
| java.naming.factory.initial | (Optional) Specifies the class name of the initial context factory. See javax.naming.InitialContext for more information. |
| java.naming.provider.url | (Optional) Specifies the provider URL property used by the InitialContext specified by the java.naming.factory.initial property. |
| java.naming.security.authentication | (Optional) Specifies the authentication mechanism to be used. Values can be one of the following:<br>• none – Use no authentication (anonymous)<br>• simple – Use weak authentication (clear-text password)<br>• space-separated list of Simple Authentication and Security Layer (SASL) mechanism names.<br>Default value is none |

# EpcisHttp Connector

The EpcisHttp connector sends events from the RFID Event Manager to the RFID Information Server using the HTTP protocol. This connecter populates the CURRENT_OBSERVATION table and the OBSERVATION_LOG table in the RFID Information Server database. Also see "Configuring the RFID Event Manager to Use an EpcisHttp Connector" on page 127.

**TABLE B-15** EpcisHttp Connector Properties

| Property | Description |
|---|---|
| classname | com.sun.autoid.logger.EpcisLogger |
| LogLevel | Defines a set of standard logging levels that can be used to control logging output. Possible values are:<br>• SEVERE (highest value)<br>• WARNING<br>• INFO<br>• CONFIG<br>• FINE<br>• FINER<br>FINEST (lowest value)<br>Also see the java.util.logging.Level and LogLevel property descriptions in TABLE 2-4. |
| EpcisUrl | Specifies the URL for the RFID Information Server. The URL is of the form:<br>http://*hostname*:*portnumber*/epcis/service<br>• The *hostname* is the name of the machine where you installed your RFID Information Server.<br>• The *portnumber* is your HTTP port. |
| HttpProxyHost | (Optional) Specifies the host name of the HTTP proxy server |
| HttpProxyPort | (Optional) Specifies the port number for the HTTP proxy server |

**TABLE B-15** EpcisHttp Connector Properties *(Continued)*

| Property | Description |
|---|---|
| `UseJms` | Specifies the delivery method. The value must be one of the following:<br>• `false`<br>• `http`<br>The default is `http`. |
| `java.naming.security.principal` | RFID Information Server user name. |
| `java.naming.security.credentials` | RFID Information Server user password. |
| `java.naming.factory.initial` | (Optional) Specifies the class name of the initial context factory. See `javax.naming.InitialContext` for more information. |
| `java.naming.provider.url` | (Optional) Specifies the provider URL property used by the `InitialContext` specified by the `java.naming.factory.initial` property. |
| `java.naming.security.authentication` | (Optional) Specifies the authentication mechanism to be used. Values can be one of the following:<br>• `none` - use no authentication (anonymous)<br>• `simple` - sue weak authentication (clear-text password)<br>• space-separated list of Simple Authentication and Security Layer (SASL) mechanism names |

APPENDIX **C**

# RFID Configuration File Reference

This appendix provides sample configuration files for the configuration of an Execution Agent. The configuration is contained in an XML configuration file for each Execution Agent. It might be necessary to create and edit these configuration files manually. By default, the RFID Event Manager is configured to run with one Execution Agent.

## Sample Default RFID Configuration Files

- The default configuration file on a Solaris system is – `/etc/opt/SUNWrfid/RfidConfig.xml`, where `/etc/opt/SUNWrfid` is the *install_config_dir*.
- The default configuration file on a Linux system is – `/etc/opt/sun/rfidem/RfidConfig.xml`, where `/etc/opt/sun/rfidem` is the *install_config_dir*.

The Execution Agent configuration syntax is illustrated in the following diagram.

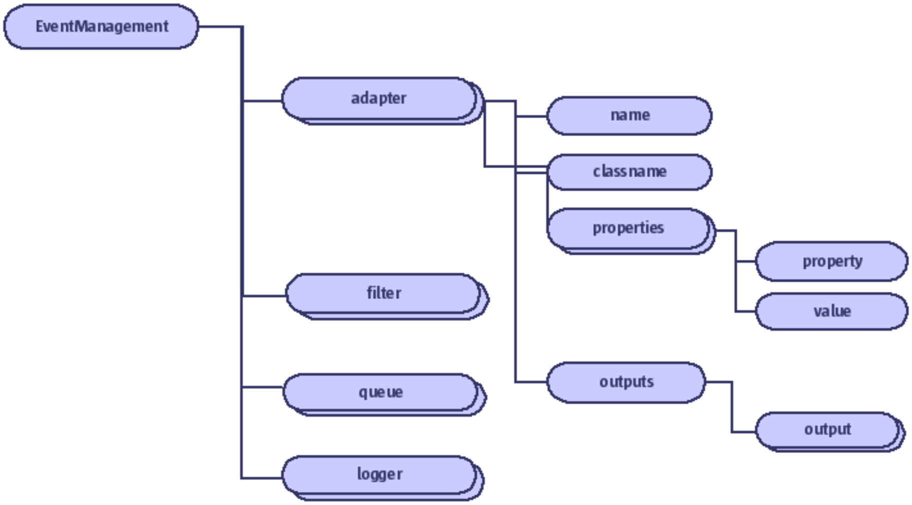


**FIGURE C-1** Configuration File Hierarchy

The name and value properties used are case sensitive. For instance, a value of `EPC` is treated differently from `Epc`.

Properties appropriate to each of the configurable entities (adapters, filters and connectors) are shown in other appendices in this book.

- Refer to Appendix A for properties information for the supported reader adapters.
- Refer to Appendix B for properties information for the supported filters and connectors (also sometimes called loggers).
- Refer to CODE EXAMPLE C-1 for an example of a typical `xml` file segment for a reader adapter.
- Refer to TABLE C-1 for an example of a typical `xml` file segment for a filter definition.

**CODE EXAMPLE C-1** Sample XML for a Reader Adapter

```
<ems:name>PMLReader</ems:name>
  <ems:classname>com.sun.autoid.adapter.pml.PMLAdapter
  </ems:classname>
     <ems:properties>
       <ems:property>LogLevel</ems:property>
       <ems:value>INFO</ems:value>
     </ems:properties>
     <ems:properties>
       <ems:property>hostname</ems:property>
       <ems:value>localhost</ems:value>
     </ems:properties>
     <ems:properties>
       <ems:property>port</ems:property>
       <ems:value>9011</ems:value>
     </ems:properties>
     <ems:properties>
       <ems:property>readerepc</ems:property>
       <ems:value>urn:epc:tag:gid-96:1.255.1</ems:value>
     </ems:properties>
     <ems:properties>
       <ems:property>autoread</ems:property>
       <ems:value>true</ems:value>
     </ems:properties>
     <ems:properties>
       <ems:property>communicationTimeout</ems:property>
       <ems:value>20000</ems:value>
     </ems:properties>
     <ems:outputs>
       <ems:output>RfidSmoother</ems:output>
       <ems:output>EpcGuiLogger</ems:output>
     </ems:outputs>
</ems:adapter>
```

**TABLE C-1** Sample `xml` Code for a Filter

```
<ems:filter>
   <ems:name>RfidSmoother</ems:name>
   <ems:classname>com.sun.autoid.filter.Smoothing</ems:classname>
      <ems:properties>
         <ems:property>LogLevel</ems:property>
         <ems:value>CONFIG</ems:value>
      </ems:properties>
      <ems:properties>
         <ems:property>MaxCycles</ems:property>
         <ems:value>5</ems:value>
      </ems:properties>
      <ems:properties>
         <ems:property>MaxPersistTime</ems:property>
         <ems:value>5000</ems:value>
      </ems:properties>
      <ems:outputs>
         <ems:output>RfidDelta</ems:output>
      </ems:outputs>
</ems:filter>
```

# Index