Perceptive Connect Runtime

Installation and Setup Guide

Version: 1.3.x

Written by: Product Knowledge, R&D
Date: February 2016

Lexmark™ Enterprise Software

Table of Contents

Overview

Perceptive Connect Runtime is the connector hub that allows you to create, configure, and enable channels that map data and functionality between Lexmark Enterprise Software products and your various business applications.

Lexmark Enterprise Software offers a variety of connectors for use with Perceptive Capture, PeopleSoft, SAP, and more. These connectors provide triggers and actions for use in customizing channels that meet your needs and bridge the gaps between your business applications. For information about available connectors, contact your Lexmark Enterprise Software representative.

This guide outlines the installation and configuration procedures for Connect Runtime. It also includes basic instructions for creating channels.

Perceptive Connect Runtime is compatible with ImageNow, version 6.7.x or higher and Perceptive Content, version 7.0.x or higher.

Prerequisites

To use Connect Runtime, you must have access to a working installation of the following software.

- Java SE Runtime Environment, version 1.8 (32-bit or 64-bit)
- A supported web browser
  - Microsoft Internet Explorer, version 10 or higher
  - Google Chrome
  - Mozilla Firefox

If you want to use Connect Runtime with Perceptive Content, you must have access to working installations of the following products.

- Perceptive Content Connector, version 1.0 or higher
- Content Integration Server, version 6.7 or higher

Overview of the setup process

To set up Connect Runtime, complete the following sections, in order.

1. Install Connect Runtime
2. Install connectors
3. Create and configure a channel

Install Connect Runtime

The following processes provide instructions for downloading and installing Connect Runtime.

Download Connect Runtime

To download the ZIP file and install Connect Runtime, complete the following steps.

1. Go to the Lexmark Enterprise Software website at www.lexmark.com and log in to the **Customer Portal**.
2. In the **Product Downloads** page, download the **Perceptive Connect Runtime for Windows ZIP** file.
3. Extract the file to the directory where you want to install Connect Runtime.

   Note If this is an upgrade, extract the file to the Connect Runtime directory you previously used. When the system prompts you to merge the Installdirectory, click **Yes**.

Install Connect Runtime

To install Connect Runtime, complete the following substeps.

1. Open a **Command Prompt** window and change the directory to **[*drive*:]\{Connect Runtime directory}.**
2. To install Connect Runtime, type `java –jar connect-runtime-installer-[version number].jar install` and press Enter.
3. To start the service, type `java –jar connect-runtime-installer-[version number].jar` start and press Enter.

   Note All data is stored in an H2 database by default. If you prefer, you can configure Connect Runtime to use Oracle, PostgreSQL, or MSSQL. See Configure Database settings to configure a database.

Upgrade Connect Runtime

You can upgrade any current version of PCR to its next minor version. At this time, it is recommended that you install all minor versions when upgrading. To upgrade Connect Runtime, complete the following steps.

1. Open a **Command Prompt** window and change to the Connect Runtime directory.
2. Type `java –jar connect-runtime-installer-[version number].jar stop` and press Enter.

3. Type `java –jar connect-runtime-installer-[`*`version number`*`].jar upgrade` and press Enter.

   Note Before the upgrade, the system creates an automatic backup. See the Roll back upgrades section for information on restoring from a backup.

4. Type `java –jar connect-runtime-installer-[`*`version number`*`].jar start` and press Enter.

   Notes

   - If you upgrade from PCR 1.0.10 to PCR 1.1.x, the `pie.mapping.path` and `org.osgi.service.http.port` properties move from the JVM Options section of the service wrapper to the **config.properties** file.
   - If you upgrade from PCR 1.2.x to 1.3.x, your existing channel mapping are migrated to the database you defined in the Configure Database settings section.
   - As of PCR 1.3, channel mappings are no longer stored on the file system, and any new mappings are stored in the database itself. As part of the upgrade to 1.3.x, the `pie.mapping.path` and `pie.mapping.backup.path` properties are removed from the **config.properties** file. They are still present in the copy of the **config.properties** file that was created with the backup prior to upgrading PCR.

About the Connect Runtime Dashboard

The Connect Runtime Dashboard includes links to the channel creation page and the Web Console where you configure connector bundles. Locate the dashboard at http://{*Connect Runtime host name*}:{*port*}. For example, http://localhost:80.

The default user name and password for the Web Console are both `admin`. To change these defaults, refer to the Apache Felix website for more information.

Note The Connect Runtime service runs on port 80 by default. To configure the port number, see the Configure the Connect Runtime port setting and path section in Appendix B.

Install connectors

About connectors

Connectors are provided as Java archive (JAR) files that you install in Connect Runtime to enable the triggers and actions they define. For installation and configuration instructions specific to the connectors you want to install, refer to the individual connector documentation.

Install a connector

To install a connector, complete the following steps.

1. In the **Perceptive Connect Runtime Dashboard**, click **Install a Connector**
2. To upload connector JAR, ZIP, or PCR files. On the **Bundle Management** page, drag the files to the **DRAG FILES HERE** area of the page.
3. Queued and processing files display in the **Pending** box. After PCR processes each item, click **Accept** to accept the installation or **Roll back** to undo the installation. You must accept or roll back the installation before PCR can process the next item.

   Notes

   - Only one user can install connectors into Perceptive Connect Runtime at any given time. After a user begins the install process, the user can acquire a lock that PCR releases after the last file installs or after 15 minutes of inactivity. If the user does not accept or roll back an installed file from the report screen, PCR automatically rolls it back when the 15 minute time limit expires.
   - The bundles in the installation may not start if any of the bundles are fragments.
   - You cannot see a report of affected channels in any of the following cases.
     - The installation contains bundles with unversioned package exports.
     - The installation contains downgrades.
     - The installation contains bundle fragments.

Bundle installation report

The bundle installation report contains the following categories.

- **Installed**. Bundles installed from the file or archive and uploaded by the user.
- **Failed**. Bundles that failed to install. This includes a message with the reason for the failure.
- **Skipped**. Files that PCR cannot install, such as non-JAR files.

Channel report

The channel report contains the following lists.

- **Fixed**. Channels that previously failed to validate that became valid after the installation.
- **Broken**. Channels that previously validated that become invalid after the installation.
- **Unknown**. Channels that PCR cannot verify as valid or invalid for the report. Usually, this is a byproduct of older channels with trigger parameters that are not stored in the database as plain text.

Verify a connector installation

To verify a connector installation, complete the following steps.

1. In **Perceptive Connect Runtime Web Console**, click **Perceptive Connect** > **View Bundles**.
2. On the **Perceptive Connect Runtime Web Console Bundles** page, there is a list of installed bundles. Verify that the connector bundles you installed are in the **Active** state. If they are not, in the **Action** column, click the bundle's **Start** button.

Configure a connector

After you install the connector, you can configure it in the View Configuration page of the Perceptive Connect Runtime Web Console. Refer to the connector documentation for specific configuration instructions.

Create and configure a channel

A channel consists of a trigger, an action, and output mapping for the resultant data. To create a channel, complete the following steps.

1. In the **Perceptive Connect Runtime Dashboard**, click **Create a channel**.
2. In the **Select a trigger** list, select a trigger, enter the data required for the trigger, and click **Next**.
3. In the **Select an action** list, select an action.

   Note If you do not want the channel to perform an action, select the **No Action** option. For example, if your channel only moves data using readers and writers, you want to select **No Action**.
4. Update the input data mapping XML for the action in the **Configure input mapping** box that appears and then click **Next**.

   Connect Runtime performs validation checks on both the input and output data mapping XML. Any errors in the XML display in the **Validation results** message under the text box.
5. In the **Configure output mapping** box, update the output data mapping XML and click **Save Channel**.
6. Click **OK** in the pop-up window if you want to enable the channel. Click **Cancel** to save the channel without enabling it.

   Important After you enable a channel, you cannot update it from the Connect Runtime Dashboard.

View configured channels

View the channel list

You can view, sort, and filter the configured channels on the Channel Listing page, which lists all of the configured channels in PCR along with basic information about each. To view the Channel Listing page, complete the following step.

- On the **Perceptive Connect Runtime Dashboard**, click **View all channels**.

Channel information

The Channel Listing page lists the following information about each configured channel.

- **ID**. Connect Runtime's ID for the channel
- **Trigger**. The name of the trigger the channel is configured to use.
- **Trigger Parameters**. The configuration value entered when initially configuring the channel.

  Note Parameters for channels created before PCR version 1.0.10 are not available until you execute or view the channel in the mapping configuration screen after upgrading. See Create and configure a channel.
- **Action**. The name of the action associated with the channel.
- **Status**. Whether the channel is enabled or disabled and satisfied or unsatisfied.
- **Configuration**. Actions that you can perform on the channel from withing the UI list.
  - Enable/disable a channel
    - An **enabled** channel is valid and can be executed by PCR.
    - A **disabled** channel is not ready for execution by PCR.
    - A channel is **unsatisfied** if the associated trigger or action are not registered with PCR. You cannot enable a channel if its state is unsatisfied.
  - View channel mapping
  - Delete a channel.

Enable or disable a channel

To enable or disable a channel from the Channel Listing page, complete the foloowingg step.

- On the **Channel Listing** page, in the **Configuration** column, click the toggle button for the channel you want to enable or disable.

View mapping configuration for a channel

You can view a channel's mapping configuration if its parameters are available. Fro a channel created in a version of PCR prior to 1.0.10, you must execute the channel before you can click its Vie wmapping button. For more information, see the Channel informationChannel information section.

To view a channel's mapping configuration, complete the following steps.

1. On the **Channel Listing** page, click the row of the channel you want to view.
2. At the top of the page, click the **View mapping** button.

   Note To view the configuration, the channel's parameters must be available. You must execute or view a legacy channel before you can click its ID. Refer to Channel information.

Clone a channel

To create a copy of a valid, existing channel and its input and output mappings, complete the following steps.

1. On the **Channel Listing** page, click the channel you want to copy.
2. At the top of the page, click **Clone Mapping**.
3. On the **Configure the channel trigger** page, enter the data the trigger requires, and click **Next**.
4. On the **Configure the channel action** page, in the **Configure input mapping** box, update the mapping if needed, and then click **Next**.
5. On the **Configure the channel** results page, in the **Configure output mapping** box, update the mapping as needed and then click **Save Channel**.
6. To enable the channel, in the dialog box, click **OK**. To save the channel without enabling it, click **Cancel**.

Delete a channel

You can delete a channel on the Channel Listing page. Note that you cannot recover a deleted channel. To delete a channel, complete the following steps.

1. On the **Channel Listing** page, click the channel you want to delete.
2. At the top of the page, click **Delete**.
3. In the confirmation dialog box, enter the name of the trigger that appears above the field, and click **Delete** to confirm the action.

Create an Experience view

As of PCR 1.3.x, support for Perceptive Experience applications (referred to in PCR as views) is built into the runtime. To create a view using Experience modules you have installed, complete the following steps.

1. In a web browser, navigate to your PCR instance.
2. Under **System Administrators**, click **Open the Web Console**.
3. Navigate to **Perceptive Connect > Configuration**
4. Under the **Perceptive Experience** category, click the “+” icon to the right of **Perceptive Experience View Configuration**.

5. Fill out the configuration for the view, taking care to select the modules you want to include, and click **Save**. A description of each of the fields is below.
    - **View entry point**: This field defines the URL to your view. The final URL will be `http://{PCR location}:{PCR port}/experience/{view name}/`
    - **Server URL**: The path to Integration Server.

      Note As of 1.0.x of the Experience Connector, a connection to Integration Server is required for authentication. However, the nature of HTTP communication prevents the required headers from being sent from one domain to another. As a result, you must set the **Server URL** field to a path relative to your Connect Runtime (such as "`/proxy/integrationserver`"). Configuring a reverse proxy (for example, an Apache server or similar) to handle communication between PCR and Integration Server is one recommended solution to this. PCR 1.3+ includes a simple REST proxy component that may be used for testing purposes but is not intended for production use.
    - **Session Keep Alive Heart Beat Interval** and **Session Timeout Seconds**: These settings are used for keeping the user's browser session active. The defaults should be sufficient for most applications.
    - **Available Modules**: All Experience modules exported by installed bundles will show up here. You should select at least one module for your view.
    - **Module configuration files**:Many Experience modules allow for end-user configuration, typically via JSON files. All such configuration files will be listed in the form. To make editing these fields easier, you may click the *Textarea* button to the right of the field to change the it to a textarea.

      Note Although all available configuration files are listed in the form, only those applicable to the modules you select in the checkboxes above will be applied.
6. The view you created in the previous step should now be available at `http://{PCR location}:{PCR port}/experience/{view name}/`

Appendix A: Upgrade connectors

Create a backup

Before upgrading a connector, we recommend that you back up your current PCR installation. This saves a snapshot of your current connector configurations that you can revert to if you encounter upgrade issues.

To create a backup of PCR, complete the following steps.

1. Open a **Command Prompt** window and change to the **Connect Runtime** directory.
2. Type `java –jar connect-runtime-installer-[`*`version number`*`].jar install backup` and press Enter.

   Note The Connect Runtime installer creates a folder called **Backup** in **Connect Runtime** directory where it stores the backup snapshot you created.

Upgrade a connector

To upgrade an installed connector, complete the process outlined in Install a connector and select the updated connector files to install. When you update the bundles, PCR overwrites the existing connector with the new connector.

Roll back upgrades

If you need to undo a connector upgrade and return to a previous installation and configuration of Connect Runtime, complete the following steps.

1. Open a **Command Prompt** window and change to the **Connect Runtime** directory.
2. Type `java –jar connect-runtime-installer-[`*`version number`*` ].jar rollback` and press Enter.
3. Type the numeral of the backup snapshot that you want to rollback to, shown in the list, and then press Enter.

Appendix B: Customize and Configure Connect Runtime

Use the methods in the following sections to troubleshoot Connect Runtime

Configure Connect Runtime logging

You use the Connect Runtime Web Console to configure the logging level and log directory. To access the console logging, complete the following steps.

1. In **Perceptive Connect Runtime Web Console**, click **Perceptive Connect > View Configuration**.
2. In the **Name** column, under **General**, click **PIF Logger**.
3. Set the **Log Level** and **Log Directory**.
4. Optional. Select either of the **Rollover Policies**.
   - If **Time** is selected, set the **Time-Based Rollover Interval**.
   - If **File** size is selected, set the **Maximum File Size (MB)**.
   - If either is selected, set the **Archive Size**.
5. Click **Save**.

   Note You do not need to restart Connect Runtime to implement the settings.

Verify Perceptive Connect Runtime service is running in Linux

You can run the following Linux command to check the status of Connect Runtime Service.

- `systemctl status perceptiveconnectruntime.service`

Run the Connect Runtime service monitor program in Windows

You can use the Connect Runtime service monitor program to view PCR properties. You can view the properties in each tab of the program menu. To implement changes, restart PCR.

To start the monitor program, complete the following steps.

1. Navigate to the **[*drive*:]\{*Connect Runtime directory*}\bin** directory.
2. Run **PerceptiveConnectRuntimew.exe**.

Connect Runtime properties

The PCR service monitor program provides the following information.

- **General**. Indicates the install path and the Startup type. Shows whether the service is running. You can stop or start the service from this tab.
- **Logging On**. Indicates the account settings.
- **Logging**. Indicates the logging level, the log path, and the log prefix. You can change the location of the log file from this table. If specified, the log directory setting in the Connect Runtime Web Console overwrites this log path.

- **Java**. Indicates the path of the Java Virtual Machine, the path of the Java Classpath, the port, memory settings, and stack size. To change the port setting, refer to Configure the Connect Runtime settings. The memory settings and stack size should remain blank.
- **Startup**. Optional. Indicates the Connect Runtime working path. Other settings in this box should remain blank.
- **Shutdown**. Optional. Settings in this box should remain blank.

Note If you get the error message "The specified service does not exist as an installed service" when launching the Connect Runtime service monitor, complete the steps in the Register the Connect Runtime service section.

Configure the Connect Runtime settings

To open the Connect Runtime configuration file, complete the following step.

- Navigate to the **Connect Runtime directory/conf** directory and open the **config.properties** file in a text editor.

Configure the port settings

The Connect Runtime service runs on port 80 by default. However, each instance of Connect Runtime on a single host machine must have a unique port number. To configure a Connect Runtime instance port number, complete the following steps.

1. In the **config.properties** file, set `org.osgi.service.http.port` to your port number. For example, if the port is 7000, the setting is `org.osgi.service.http.port=7000`.
2. Save the file and restart the runtime.

Configure Database settings

Perceptive Connect Runtime stores channel mappings into the database defined by these settings. In case these database settings are omitted, any PCR data is persisted in the in-memory H2 database provided through PCR.

Perceptive Connect Runtime supports the following databases: H2, MSSQL, PostgreSQL, and Oracle.

Modify the following properties to the **config.properties** file.

- `pcr.db.server.type=[serverType]`
- `pcr.db.servername=[serverName]`
- `pcr.db.port=[portNumber]`
- `pcr.db.databasename=[databaseName]`
- `pcr.db.username=[username]`
- `pcr.db.password=[password]`

Where

- `serverType` is the type of database PCR will be connecting to with h2, `mssql`, `oracle`, and `postregresql` as options.
- `serverName` is the IP address or DNS name of your database server.

- `portNumber` is the open port on which your database is listening.
- `databaseName` is the name of the database that PCR will use to store all of its data.
- `username` is the database user with access to the database.
- `password` is the database required password.

Example

```
pcr.db.server.type=mssql
pcr.db.servername=PCR_DB_SERVER
pcr.db.port=1433
pcr.db.databasename=PCR_DATA
pcr.db.username=pcrUser
pcr.db.password=imagenow
```

Notes

- By default, when Perceptive Connect Runtime is installed, H2 is the chosen database and these properties are not required.
- For MSSQL, PostgreSQL, or Oracle Database, a database must be created to store PCR data, which can be named anything.
- For MSSQL, PostgreSQL, or Oracle Database, a user must have privileges to update, create, and delete tables in the PCR database.

Configure the session timeout

The default timeout for the Connect Runtime is 20 minutes. To change this time frame, complete the following steps.

1. In the **config.properties** file, change the value of `org.apache.felix.http.session.timeout` to a duration in minutes, such as `org.apache.felix.http.session.timeout=60`.
2. Save the file and restart the runtime.

Configure the base URL for the web console

By default, the Connect Runtime web console can be found at http://{Connect Runtime host name}:{port}/. To change this URL, complete the following steps.

1. In the **config.properties** file, set `felix.webconsole.manager.root` to the new URL, similar to the following example: `felix.webconsole.manager.root=/webconsole`.

   Note The Connect Runtime dashboard exists at the root of your path, such as *http://{Connect Runtime host name}:{port}/*, so you cannot use the path / for this setting.
2. Save the file and restart the runtime.

Additional Configuration for Linux

To open the Connect Runtime linuxEnvironmentFile configuration file, complete the following step.

- Navigate to the **Connect Runtime directory/conf** directory and open the **linuxEnvironmentFile** in a text editor.

To customize java location

1. In the **linuxEnvironmentFile** file, set `PCR_JAVA_BIN` to the path for the java bin you would like to run. By default, the java used will be the java found in your Linux path. For example, `PCR_JAVA_BIN='/bin/java'`.
2. Save the file and restart the service.

To add any additional JVM arguments

1. In the **linuxEnvironmentFile** fie, set `PCR_JVM` to include any JVM arguments you want to pass to Connect Runtime. For example, `PCR_JVM='-Xdebug – Xrunjdwp: transport=dt_socket, server=y,suspend=n,address=1044'`.
2. Save the file and restart the service.

About configuration settings with file paths

Java Properties uses the config.properties file. As a result, all settings that contain file paths should use forward slashes (/) and not backslashes (\). For more information, refer to Java Properties documentation on the Oracle website.

Uninstall Connect Runtime

The uninstall action removes the Connect Runtime service from Windows services. It does not delete the installation directory. To uninstall Connect Runtime, complete the following steps.

1. Open a **Command Prompt** window and change the **Connect Runtime directory**.
2. Type `java –jar connect-runtime-installer-[`*`version number`*`].jar uninstall` and press Enter.

Register the Connect Runtime service

The installer automatically registers the Connect Runtime Service with Windows when installing. However, it may be necessary to re-register the service, particularly when moving the Connect Runtime to another location on the disk. To re-register the service, complete the following steps.

1. Open a **Command Prompt** window and change to the **Connect Runtime directory**.
2. Type `java –jar connect-runtime-installer-[`*`version number`*`].jar install register` and press Enter.

Unregister the Connect Runtime Service

You can unregister the Connect Runtime service without uninstalling Connect Runtime. To unregister the service, complete the following steps.

1. Open a **Command Prompt** window and change the **Connect Runtime directory**.
2. Type `java –jar connect-runtime-installer-[version number].jar uninstall` and press Enter.

Appendix C: Actions

An action is a connector-defined task configured in the channel. The action executes when the channel is triggered.

Perceptive Connect Runtime provides the ability, by default, to select No Action as an option of any connector you install on Connect Runtime. This interface is available for use in connector channels where you only use readers and writers. It does not perform any logic.

For more information about other channels actions, see the installation guide for any Perceptive connector.

Appendix D: Readers

Readers are components you use to configure connector channels to retrieve values from other applications. The channel uses these values in the data context during its execution. You use readers to configure the action input mapping. Input mappings allow a channel to have the required data to execute the action. You can invoke readers using specific XML tags, which Perceptive Connect defines per reader.

Some readers, such as the Date transformer and the XML transformer, take a set of data and transform it.

Note The channel data context refers to the data from various sources that is available in the channel for action input and results output configuration. The data available in the context depends on the trigger the channel uses as well as the connectors that are installed in Connect Runtime.

Perceptive Connect Runtime provides the following readers for use with any connector.

Date Format reader

The Date Format reader is used to transform date strings between formats, such as mm/dd/yyyy and yyyy-dd-mm. The reader takes a date string from the parameter reference, transforms the format, and outputs the resulting date string. The Date Format reader provides the following configuration fields.

- `dateReference` specifies a mapped string parameter that represents a date.
- `inputFormat` specifies the dateReference (input) date format.
- `outputFormat` specifies the output date format.

To use the Date Format reader, use the following XML format within a parameter.

```
<c:dateFormatter>
       <c:dateReference></c:dateReference>
       <c:inputFormat></c:inputFormat>
       <c:outputFormat></c:outputFormat>
</c:dateFormatter>
```

The date format strings should follow the patterns used by Java `SimpleDateFormat` class. Refer to the Oracle website for more information.

Example

```
<c:parameter>
       <c:name>dateInput</c:literal>
</c:parameter>
<c:parameter>
       <c:name>formattedDate</c:name>
       <c:dateFormatter>
              <c:dateReference>dateInput</c:dateReference>
              <c:inputFormat>yyyy-dd-mm</c:inputFormat>
              <c:outputFormat>mm/dd/yyyy</c:outputFormat>
       </c:dateFormatter>
</c:parameter>
```

In this example, the Date Format reader transforms the `dateInput` value of `2014-08-12` to an output of `12/08/2014`.

List Item reader

The ListItem reader lets you access a single value from a list of items provided by a trigger or a reader.

To use the reader, include the following XML template in the channel input mapping, complete with appropriate values.

```
<parameter>
        <name></name>
        <listItem>
                <listRef></listRef>
                <index></index>
        </listItem>
</parameter>
```

The `listRef` field contains a reference to the list item you want to access. The index field specifies the location of the list item in the index. Index numbers start at "0."

Positive Index

```
<parameter>
        <name>myItem</name>
        <listItem>
                <listRef>myList</listRef>
                <index>3</index>
        </listItem>
</parameter>
<c:parameter>
        <c:name>myList</c:name>
        <c:trigger>Results</c:trigger>
</c:parameter>
```

In the example, the reader retrieves the item at the specified index number, from the list `listRef` associated with the `index` number, and stores it in the context as `myItem`.

Negative Index

```
<parameter>
        <name>myItem</name>
        <listItem>
                <listRef>myList</listRef>
                <index>-2</index>
        </listItem>
</parameter>
<c:parameter>
        <c:name>myList</c:name>
        <c:trigger>Results</c:trigger>
</c:parameter>
```

In the example, the reader retrieves the item at the specified index number, from the list `listRef` associated with the `index` number. The negative index number tells the reader to retrieve the second-to-last element from the list and stores it in the context as `myItem`.

Literal reader

The Literal reader reads a literal string and then stores the value in the data context for further use.

To use the Literal reader, enter the following XML within a parameter.

```
<c:literal></c:literal>
```

Example

```
<c:parameter>
       <c:name>eFormName</c:name>
       <c:literal>AP Invoice</c:literal>
</c:parameter>
```

In this example, the Literal reader reads and stores the value `AP Invoice` under the name `eFormName`. You can reference `eFormName` later in the data context, as needed, to retrieve the value `AP Invoice`.

Regular Expression transformer

The Regular Expression transformer completes a RegEx find and replace operation on a string. It provides the following configuration fields.

- `find` is the string or regular expression for which it searches.
- `replace` is the string or regular expression to replace the matched text.

You should format both fields in a manner that the Java Pattern class's compile() method can accept.

To use the regular expression transformer, use the following XML format.

```
<c:regexTransform>
       <c:find></c:find>
       <c:replace></c:replace>
</c:regexTransform>
```

Example

```
<c:parameter>
       <c:name>newDepartmentName</c:name>
       <c:regexTransform>
              <c:reference>departmentName<c:/reference>
              <c:find>R&amp;D</c:find>
              <c:replace>AP/ERP</c:replace>
       </c:regexTransform>
</c:parameter>
```

In this example, the Regular Expression transformer finds the `departmentName` item in the data context and then searches for the string "R&D." Then it replaces and all occurrences with the phrase "AP/ERP." The transformer stores the result in the `newDepartmentName` context item for use during the channel's execution.

Example

```
<c:parameter>
      <c:name>trimmedText</c:name>
      <c:regexTransform>
            <c:reference>rawText</c:reference>
            <c:find>^\s*(.*?)\s*$</c:find>
            <c:replace>$1</c:replace>
      </c:regexTransform>
</c:parameter>
```

In this example, the Regular Expression transformer removes any leading or trailing whitespace from the `rawText` data context. The transformer stores the result in the `trimmedText` context item for use during the channel's execution.

Stream to XML reader

The `StreamToXML` reader takes a reference to a `MappingInputStream` object and transforms it into an `org.w3c.dom.Document` object.

To transform a `MappingInputStream` into an `org.w3c.dom.Document` object, include the following XML template in the channel input mapping, complete with the appropriate values.

```
<c:streamToXML>
      <c:streamRef></c:streamRef>
</c:streamToXML>
```

The `streamRef` field is a reference to the `MappingInputStream`.

Example

```
<c:parameter>
      <c:name>xmlDoc</c:name>
      <c:streamToXML>
            <c:streamRef>file</c:streamRef>
      </c:streamToXML>
</c:parameter>
<c:parameter>
      <c:name>file</c:name>
      <c:trigger>FileStreamParam</c:trigger>
</c:parameter>
```

In the example, `trigger` provides a MappingInputStream parameter called `FileStreamParam`. FileStreamParam is stored in the context as `file`, which is passed in to the `streamToXML` reader. The parsed XML Document is then stored in the context as `xmlDoc`. The XML reader or XML transformer can then use the parameter.

Trigger reader

The Trigger reader reads values in the data context that were provided as outputs by the channel's trigger.

To use the Trigger reader, enter the following XML code.

```
<c:trigger></c:trigger>
```

For example, the Perceptive Intelligent Capture Connector provides the Export trigger. This trigger provides the Content document ID as an output called `DocumentId`. To read that output value and store it in the data context for further use with the reference `DocId`, enter the following XML code.

```
<c:parameter>
      <c:name>DocId</c:name>
      <c:trigger>DocumentId</c:trigger>
</c:parameter>
```

XML reader

The XML reader reads values from an XML document. The reader processes repeating elements and single elements. The XML reader provides the following configuration fields.

- `reference` A configuration field specifying the data context value that contains the XML document. If the `xmlRowSource` node is not nested under another `xmlRowSource` node, you must use this field. Otherwise, it is optional.
- `context` A configuration field specifying the parent element. If the `xmlRowSource` node is nested inside another `xmlRowSource` node, you must use this field. Otherwise, it is optional.

  Note You must use either the **reference** or **context** field in each `<c:xmlSource>` or `<c:xmlRowSource>` instance.
- `id` A configuration field setting a unique value for `xmlSource` and `xmlRowSource` children to use when specifying their `context` element.
- `xpath` A field that specifies the Xpath location, in the XML document, of the values to read.

To read repeating elements, you would use an `xmlRowSource` node and the `reference` or `context`, `id`, and `xpath` fields. Use the following XML format.

```
<c:xmlRowSource>
      <c:reference></c:reference> OR <c:context></c:context>
      <c:id></c:id>
      <c:xpath></c:xpath>
</c:xmlRowSource>
```

To read a single element from an XML document, you use an `xmlSource` node and the `reference` or `context` and `xpath` fields. Use the following XML format.

```
<c:xmlSource>
      <c:reference></c:reference> OR <c:context></c:context>
      <c:xpath></c:xpath>
</c:xmlSource>
```

The following examples assume there is a trigger that provides an XML document. The Trigger reader names the provided document “XMLDoc” in the data context. Then this component reads the value at “IntelligentCaptureDocument/InvHeader/INVOICE_Number” from the referenced XML document.

Single element

```
<c:parameter>
    <c:name>XMLDoc</c:name>
    <c:trigger>XmlDocument</c:trigger>
</c:parameter>
<c:parameter>
    <c:name>InvNum</c:name>
    <c:xmlSource>
        <c:reference>XMLDoc</c:reference>
        <c:xpath>IntelligentCaptureDocument/InvHeader/INVOICE_NUMBER</c:xpath>
    </c:xmlSource>
</c:parameter>
```

Repeating elements

```
<c:parameter>
    <c:name>XMLDoc</c:name>
    <c:trigger>XmlDocument</c:trigger>
</c:parameter>
<c:rowset>
    <c:name>page</c:name>
    <c:xmlRowSource>
        <c:reference>XMLDoc</c:reference>
        <c:id>Root</c:id>
        <c:xpath>/IntelligentCaptureDocument</c:xpath>
    </c:xmlRowSource>
    <c:mapping>
        <c:parameter>
            <c:name>InvNumber</c:name>
            <c:xmlSource>
                <c:context>Root</c:context>
                <c:xpath>InvHeader/INVOICE_NUMBER</c:xpath>
            </c:xmlSource>
        </c:parameter>
    </c:mapping>
</c:rowset>
```

XML transformer

The XML transformer takes a set of data and transforms it into a provided XML template or schema. The XML transformer provides the following three configuration fields.

- `data` A reference to a `ContextList` in the data context. The value used in the data field needs to be a rowset of values from the XML reader.
- `template` The XML document into which the XML transformer writes and saves the data. Any repeating elements in the template only need to have one instance or the exact number of instances found in `data`. Otherwise, the component does not work.
- `mapping` A list of key and value pairs that override the transformation location of some values in the `data` element.

To use the XML transformer, use the following XML format.

```
<c:xmlTransform>
    <c:data></c:data>
    <c:template></c:template>
    <c:mapping>
        <c:entry>
            <c:key></c:key>
            <c:value></c:value>
        </c:entry>
        ...
    </c:mapping>
</c:xmlTransform>
```

For a complete example of how to use the XML transformer in your channel, refer to the XML example provided with your connector.

Appendix E: Configuring SSL

Perceptive Connect Runtime has full support for SSL/TLS, including client-side authentication. Use the following methods to configure SSL.

About the config.properties file

The config.properties file contains the configuration settings for PCR. You can find it in the [connect install directory]/conf/ directory. Note that when modifying settings in this file, you cannot have any trailing spaces.

Configure inbound SSL connections

To configure inbound SSL connections, complete the following steps.

1. In the **[*connect install directory*]/conf/** directory, open the **config.properties** file with a text editor.
2. Add the inbound SSL connections settings, replacing **[*connect install directory*]** and **[*password*]** with the path and password used in your keystore. You can configure the `org.osgi.service.http.port.secure` setting to any valid port.
3. Save and close the **config.properties** file.
4. Optional. To test the configuration, sue a browser to connect to the PCR Dashboard on the SSL port you configure. For example, https://myserver:443.

Example

```
org.apache.felix.https.enable=true
org.osgi.service.http.port.secrure=443
org.apache.felix.https.keystore=[connect install directory]/data/pcr-keystore.jks
org.apache.felix.https.keystore.password=[password]
org.apache.felix.https.keystore.key.passwrod=[password]
org.apache.felix.https.truststore=[connect install directory]/data/pcr-truststore.jks
org.apachhe.felix.https.truststore.password=[password]
```

Inbound SSL connections

Perceptive Connect Runtime supports the following inbound SSL connections in the config.properties file. For more information, refer to the Apache Felix HTTP Service documentation.

- `org.apache.felix.https.enable` Enables SSL in the runtime. Options are TRUE and FALSE.
- `org.osgi.service.https.port.secure` The port used to accept SSL requests. Use any valid port number.
- `org.apache.felix.https.keystore` The path for the PCR keystore. PCR may have its own keystore, or share a keystore with other applications on the system.
- `org.apache.felix.https.keystore.password` The password for the keystore.
- `org.apache.felix.https.keystore.key.password` The password for the key (alias) in the keystore.

  Note By default, this is the same as `org.apache.felix.https.keystore.password`, but the two passwords do not have to match.

- `org.apache.felix.https.truststore` The path for the PCR truststore.
  - PCR may have its own truststore or share a truststore with other applications on the system. For example, you might use the default Java truststore in the **[*JAVA_HOME*]/lib/security/cacerts** file. You can find more information on the Oracle website.
- `org.apache.felix.https.truststore.password` The password for the truststore. This must be the same as the `org.apache.felix.https.keystore.password` setting.

Configure outbound SSL connections

For outbound connections, PCR uses the Java Secure Socket Extension (JSSE) truststore. By default, the JSSE trust store is located in the Java install directory. It is recommended to configure the Connect Runtime to use the truststore you specified for the inbound SSL connections. To configure the JSSE truststore, complete the following steps.

Windows configuration

1. Navigate to **[*connect install directory*]/bin/**.
2. Run **PerceptiveConnectRuntimew.exe**.
3. On the **Java** tab, in the **Java Options** section, add the following setting.

   `Djavax.net.ssl.trustStore=[*connect install directory*]/data/pcr-truststore.jks`. Specify the file configured for the `org.apache.felix.https.truststore` setting in the **config.properties** file.

   Note Replace the **[*connect install directory*]** with the path to the connect installation.

Linux configuration

1. Navigate to the **[Connect Runtime directory]/conf** directory and open the **linuxEnvironmentFile** file in a text editor.
2. Add or modify the PCR_JVM by setting `PCR_JVM='-Djavax.net.ssl.trustStore=[*connect install directory*]/data/pcr-truststore.jks'`. Specify the file configured for the `org.apache.felix.https.truststore` setting in the **config.properties** file.

Configure client authentication

To enable client validation, complete the following steps.

1. Navigate to **[*connect install directory*]/conf/**.
2. Open the **config.properties** file with a text editor.
3. Add `org.apache.felix.https.clientcertificate=needs`.
4. Configure clients to provide appropriate certificates.

   Note The exact steps required depend on the client. Most browsers either are configured in the OS or have their own configuration for client certificates. Other web services have their own configuration.
5. If a client attempts to connect without a valid certificate, an error such as the following occurs in the **pif.all.log** file.

```
2014-11-06 10:30:19 [o.e.jetty.io.nio] WARN –
Javax.net.ssl.SSLProtocolException: handshake alert: no_certi
```

Appendix F: Metrics

PIF Metrics allow you to capture performance data for the framework as well as individual connectors. Included are several built-in categories for measuring metrics over varying amounts of time. PIF Metrics writes output data as a CSV file in a user-defined location.

Configure Metrics

By default, Connect continuously collects and writes metrics to a file over a four-hour period. To configure metrics, complete the following steps.

1. In the **Perceptive Connect Runtime Dashboard**, click **Open the Web Console**.
2. In the **Perceptive Connect Runtime Web Console**, click **Perceptive Connect > View Configuration**.
3. In the **Name** column, under **General**, click **PIF Metrics**.

Metrics settings

Connect Metrics include the following settings.

Metrics Directory. The path to the directory for writing Connect Metrics.

Metrics Categories. The Metric categories to report.

Continuous Collection. If enabled, Connect collects metrics on a continuous basis for the selected categories as long as Connect Runtime runs. The default setting is enabled. This setting overrides the `Collection Period` setting.

Interval Period. The period of time for which Metrics aggregates data per category. For example, if you set `Interval Period` to 1 with `the Interval Period Unit` as hours, and execute channels throughout that hour, there would be a single listing of the "Channel Execution" category with aggregate data for that hour. When an hour has passed from this entry, metrics data is collected and aggregated again towards the next entry.

Interval Period Unit. The unit of time for the interval period.

Collection Period. Defines the total duration of time for which Connect collects metrics. Connect does not use this setting if `Continuous Collection` is enabled.

Collection Period Unit. The unit of time for the collection period.