Endeca® Content Acquisition System

Quick Start Guide

Version 3.0.1 • December 2011

Contents

Copyright and disclaimer

Trademarks

Chapter 1

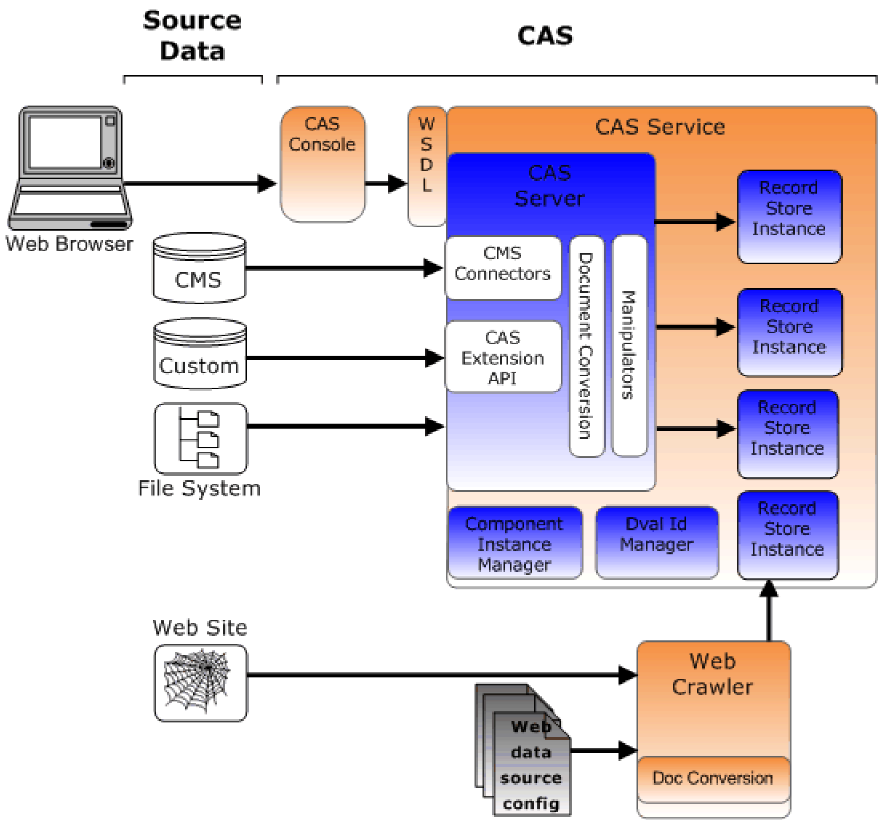
Using the Endeca Content Acquisition System

The CAS Quick Start Guide describes the basics of the Endeca Content Acquisition System (CAS) and then walks you through the high-level process of installing Endeca with CAS, adding manipulators, crawling data sources, and processing the Endeca records in a Forge pipeline. This scenario describes how to get started with a new installation: it does not describe an upgrade or migration.

Overview of the Endeca Content Acquisition System

The Endeca Content Acquisition System is a set of components that add, configure, and crawl data sources for use in an Endeca application. Data sources include file systems, content management systems, Web servers, and custom data sources. The Endeca Content Acquisition System crawls data sources, converts documents and files to Endeca records, and stores them for use in an Forge pipeline.

The following image shows the Endeca Content Acquisition System components as they work together in a typical implementation to crawl data sources and produce Endeca records:



The Endeca Content Acquisition System is made up of the following components:

- The Endeca CAS Service is a servlet container that runs the CAS Server, the Component Instance Manager, and any number of Record Store instances (one per data source).
- The CAS Server is the component that manages all file system and CMS crawling operations. The CAS Server is documented in the *Endeca CAS Developer's Guide*.
- The CAS Console for Endeca Workbench is a Web-based application used to crawl various data sources including file systems and content management systems. During the Content Acquisition System installation, the CAS Console is installed as an extension to Endeca Workbench. The CAS Console is documented in the *Endeca CAS Console Help*.
- The CAS Server API allows users to write programs that communicate with the CAS Server. The CAS Server API has a WSDL interface and also a CAS Server Command-line Utility. The API is documented in the *Endeca CAS API Guide*.
- The Dimension Value Id Manager is a CAS component that creates, stores, and retrieves dimension value identifiers.
- The Endeca Web Crawler manages all Web crawl-related operations. This component is documented in the *Endeca Web Crawler Guide*.
- Endeca CMS connectors are available for use in the CAS Console for Endeca Workbench or the CAS Server API. CMS connectors provide a means to access and crawl data sources in a wide variety of CMS types, such as Documentum, eRoom, FileNet, JSR-170 compliant repositories, Lotus Notes, Microsoft SharePoint, and Interwoven TeamSite.
- The Component Instance Manager creates, lists, and deletes Record Store instances. The Component Instance Manager has a WSDL interface and also a CIM Command-line Utility.
- The Endeca Record Store provides persistent storage for generations of records. The Record Store has a WSDL interface and also a Record Store Command-line Utility. The CAS Server writes crawl output from each data source to a unique Record Store instance.

- The CAS Extension API provides interfaces and classes to build extensions such as custom data sources and custom manipulators. You package extensions into a plug-in and install it into the Content Acquisition System. After you install the plug-in, the extensions are available and configurable using the CAS Console, the CAS Server API, and the CAS Server Command-line Utility.

Installing the required Endeca components

You must install the following Endeca components in order to use the Content Acquisition System.

1. Install Endeca MDEX Engine 6.1.x. (This is available under `Product Downloads > Endeca IAP Solution 6`.)
2. Install Platform Services 6.1.x (This is available under `Product Downloads > Endeca IAP Solution 6`.)
3. Install Developer Studio 6.1.x. (This is available under `Product Downloads > Endeca IAP Solution 6`.)
4. Install Endeca IAP Workbench 2.1.x. (This is available under `Product Downloads > Endeca IAP Solution 6`.)
5. Install the Deployment Template 3.2. (This is available under `Tools and Utilities`.)
6. Install the Content Acquisition System 3.0.x. (This is available under `Product Downloads`.)

Note: After installing CAS, you must also upgrade the Deployment Template. For details, see "Updating the Deployment Template to use the WSDL client stubs and the CAS component" in the *Endeca CAS Installation Guide*.

Creating an application with the Deployment Template

You create an application by running the `deploy` script in `\deploymentTemplate-3.2\bin`.

To create an application:

1. On your local file system, create an empty folder to store the new application.

   This folder is typically named the `Apps` directory under `Endeca`, for example `C:\Endeca\Apps\`.
2. Open a Command Prompt window, change directory to `\Solutions\deploymentTemplate-3.2\bin`, and run the `deploy` script.
   For example:
   ```
   C:\Endeca\Solutions\deploymentTemplate-3.2\bin>deploy
   ```
3. Provide configuration parameters as the `deploy` script prompts you. In most cases you should take the defaults as follows:
   - Confirm the Endeca IAP version.
   - Select `1` for a Dgraph deployment.
   - Name the application (for example, `TestApp`).
   - Specify the empty folder you created in step one (for example, `C:\Endeca\Apps`).
   - Specify the EAC port (use the default `8888`).
   - Specify `Y` to integrate the application with Endeca Workbench.

- Specify the Workbench port (use the default `8006`), the Dgraph port (use the default `15000`), the redundant Dgraph port (`15002`), and the log server port (`15010`).

At the end of the process, you see the message:

```
05/22/2009 13:36:29 [deploy.pl] INFO:  Processing install with id 'Dgraph'
05/22/2009 13:36:31 [deploy.pl] INFO:  Application successfully deployed.
```

Also, if you go to the application directory, you will find a Developer Studio project for CAS named `cas_crawl_pipeline`. You can use the project later as a starting point to build your custom Forge pipeline, which is described in *Processing the Endeca records in a baseline update or a partial update* on page 15. In the above example, this project is located in `C:\Endeca\Apps\TestApp\config\cas_crawl_pipeline`.

Provisioning an application with the EAC Central Server

After creating an application, you provision it with the EAC Central Server by running the `initialize_services` script. Provisioning the application makes it available for configuration in Endeca Workbench.

To provision an application:

Open a Command Prompt window, change directory to the `\control` folder of the application you created earlier, and run the `initialize_services` script.
For example:

```
C:\Endeca\Apps\TestApp\control>initialize_services.bat
```

There are no parameters to provide while the script runs. At the end of the process, you see the following information:

```
[05.22.09 16:45:51] INFO: Finished updating IAP Workbench.
Finished updating EAC.
```

Creating extensions to CAS

If you want to access custom data sources or manipulate Endeca records during a crawl, you can create and install CAS extensions. Extensions include data sources and manipulators.

In most cases, a plug-in developer creates CAS extensions, packages them as a plug-in, and distributes the plug-in to a CAS developer. For details about creating and packaging CAS extensions, see the *CAS Extension API Guide*.

Installing a plug-in into CAS

After receiving a plug-in (a JAR or set of JAR files) from a plug-in developer, a CAS application developer installs the plug-in into CAS.

The Content Acquisition System detects each plug-in and validates the extensions within it by checking the uniqueness of extension IDs and by checking for the presence of an annotation of either `@Cas¬DataSource` or `@CasManipulator` for each extension.

To install a plug-in into CAS:

1. Stop Endeca CAS Service.
2. Navigate to `<install path>\CAS\`*`version`*`\lib\cas-server-plugins` and create a *`plugin-name`* subdirectory for each plug-in.
   For example: `CAS\`*`version`*`\lib\cas-server-plugins\JDBCDataSourceExt`
3. Copy the plug-in JAR or JARs, and any dependent JAR files, to `<install path>CAS\`*`version`*`\lib\cas-server-plugins\`*`plugin-name`* .
4. Repeat the steps above as necessary for multiple plug-ins.
5. Start Endeca CAS Service.

You can confirm that an extension is installed by runing the `listModules` task of the CAS Server Command-line Utility and specifying a `moduleType` of either `SOURCE` or `MANIPULATOR`. The task returns the installed modules. For example, this task shows that a custom data source named `Sample Data Source for testing` is installed:

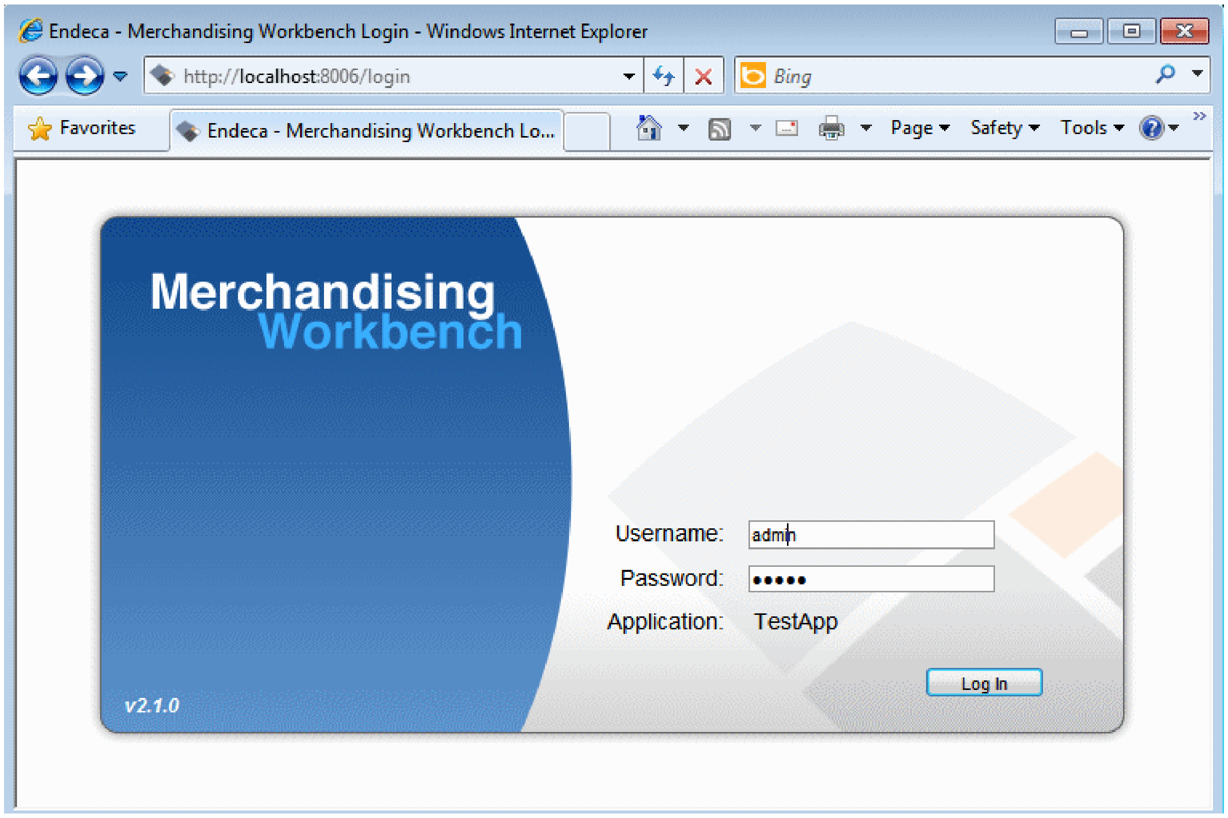
```
C:\Endeca\CAS\3.0.0\bin>cas-cmd listModules -t SOURCE
Sample Data Source
 *Id: Sample Data Source
 *Type: SOURCE
 *Description: Sample Data Source for testing

File System
 *Id: File System
 *Type: SOURCE
 *Description: No description available for File System
 *Capabilities:
   *Binary Content Accessible via FileSystem
   *Data Source Filter
   *Has Binary Content
   *Expand Archives
```

Adding a data source using CAS Console

Use CAS Console for Endeca Workbench to access your deployed application, then add a data source and configure its settings.

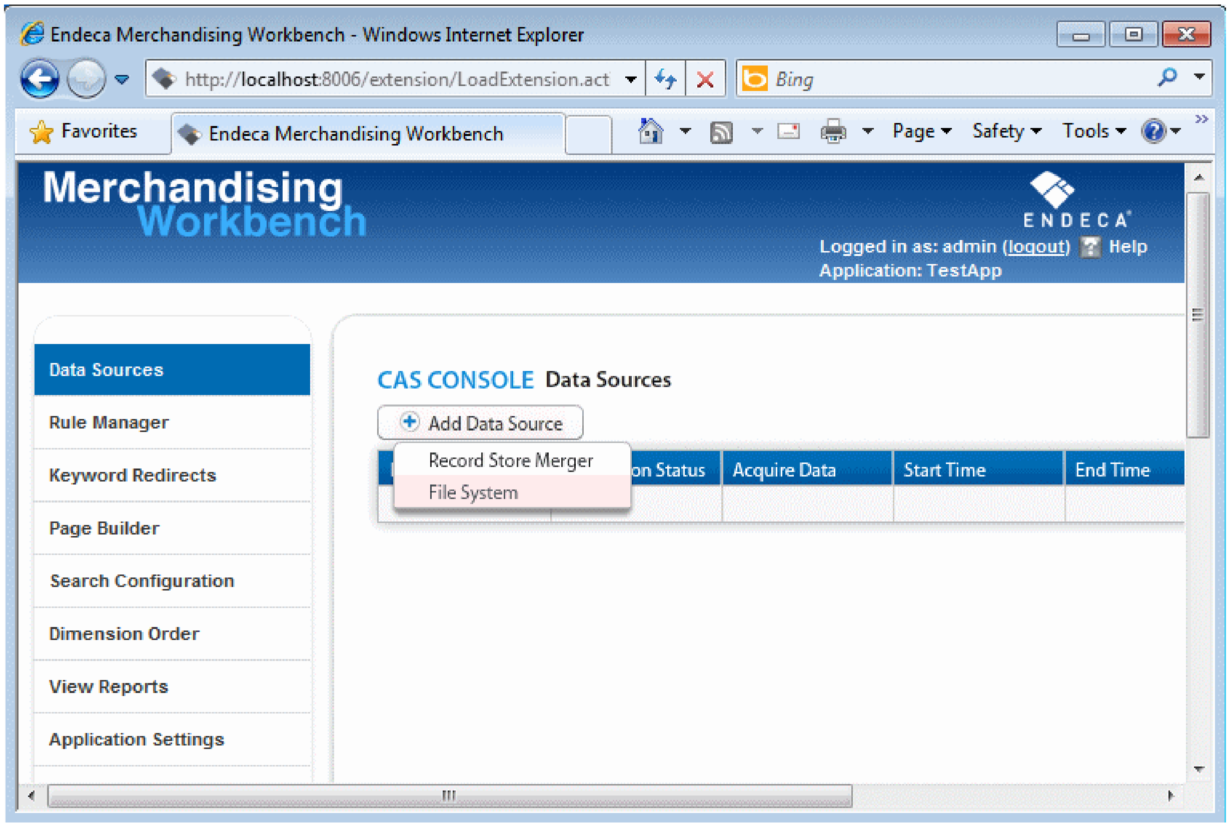
To add a data source using CAS Console:

1. Start a Web browser and run Endeca Workbench.
   For example, if you accepted the installation defaults, Endeca Workbench is running on `http://`*`host`*`:8006/login`.
2. Provide **Username** and **Password** values. You can use the default `admin/admin` login.
3. From the **Application** list, select the application you previously created using the `deploy` script, or if you have one application deployed, it will be selected by default.
   For example:


Endeca - Merchandising Workbench Login - Windows Internet Explorer
http://localhost:8006/login
Bing
Favorites
Endeca - Merchandising Workbench Lo...
Page
Safety
Tools
Merchandising Workbench
v2.1.0
Log In

Username: admin
Password: •••••
Application: TestApp

4. Click **Log In**.
5. Select the **Data Sources** page, click **Add Data Source**, and select an appropriate data source. For example:



6. Specify data source filters and advanced settings as necessary.

   For details about configuring either of these data sources, see the *Endeca CAS Console Help*.
7. Click **Save**.

Adding a manipulator to a data source

If necessary, you can manipulate Endeca records by adding and configuring a manipulator. When you start an acquisition, manipulators also run and perform record modification.

Before you can perform this task, a plug-in developer must create a manipulator and a CAS developer must install the manipulator.

To add a manipulator to a data source:

1. Start Endeca Workbench. and
2. Log in to the application that contains the data source you want to modify.
3. Select the **Data Sources** page and click a data source name to access its acquisition steps.
4. Click **Add Manipulator...**
5. Select a manipulator and click **Ok**.

   The **Edit Manipulator** page displays.
6. Specify configuration properties as necessary for the manipulator.

To determine the configuration property values, you may have to coordinate with the extension developer who created the manipulator, or you can run the `getModuleSpec` task of `cas-cmd` to retrieve the configuration properties of a manipulator.

7. Click **Save**.
8. Add additional manipulators as necessary.

Crawling data sources

You can crawl data sources using any of the following: the CAS Console, the CAS Server Command-line Utility, and from the CAS Server API.

Crawling from the CAS Console for Endeca Workbench

The **Data Sources** page of CAS Console displays all data sources available for crawling. You can start crawling a particular data source by clicking **Start** in the **Acquire Data** column. For further information, see the *Endeca CAS Console Help*.

Crawling from the CAS Server Command-line Utility

You can start and stop a crawl from the CAS Server Command-line Utility by running either the `startCrawl` or `stopCrawl` tasks. For further information, see the chapter in the *Endeca CAS Developer's Guide* on the CAS Server Command-line Utility.

Crawling from the CAS Server API

You can start a crawl by calling the `CasCrawler.startCrawl()` method from an application. For further information, see the *Endeca CAS API Guide*.

Crawling using Deployment Template Scripts

In a typical production environment, you control the application and CAS crawls using the Deployment Template. This requires integrating CAS into the Deployment Template scripts. For details, see "Integrating and Running CAS Crawls" in the *Deployment Template Usage Guide*.

No matter how you crawl a data source, the result is a set of Endeca records that is stored in a Record Store instance per data source. These records can be processed in a Forge pipeline.

Crawling a Web server

You crawl a Web server data source using the Endeca Web Crawler. The Endeca Web Crawler is shipped with the Content Acquisition System; however, it is configured and run as a stand-alone component outside the Endeca CAS Service.

To crawl a Web server data source:

1. Configure the Web Crawler to crawl for one or more Web server data sources.

   For details, see the *Endeca Web Crawler Guide*.
2. For operational simplicity in your Forge pipeline, re-configure the Web Crawler to write record output to a Record Store instance. (By default the Web Crawler writes record output to a record output file.)

For details, see "Configuring Web crawls to write output to a Record Store instance" in the *Endeca Web Crawler Guide*.

Processing the Endeca records in a baseline update or a partial update

The procedures described in this section involve pipeline development within Developer Studio, which may be installed on a separate machine from your CAS installation.

1. Create or revise a Developer Studio project (a baseline update or partial update).

   For details about baseline updates, see the *Endeca Developer Studio Help* and also see "Creating a Pipeline to read Endeca records" in the *CAS Developer's Guide*.

   For details about partial updates, see the *Partial Updates Guide.*
2. Create a custom record adapter in the Forge pipeline to read the Endeca records from one or more Record Store instances. You can create a record adapter to read from a single Record Store instance or you can create a record adapter to read from multiple Record Store instances.
3. Incorporate the Forge pipeline into your Deployment Template.

   For details, see the *Deployment Template Usage Guide*.
4. Run the baseline update or partial update pipeline.

   The update runs Forge, Dgidx, and produces indexed Endeca records for use in the MDEX Engine.

   For details, see the *Deployment Template Usage Guide*.

Creating a record adapter to read from one or more Record Store instances

By default, the CAS writes output from a CMS, file system, or custom data source to a Record Store instance. The Web Crawler can also be configured to write output to a Record Store instance. If an application contains multiple data sources, there are multiple Record Store instances that result. Forge can read the Endeca records from any number of Record Store instances using a custom record adapter.

You configure a custom record adapter in Developer Studio with Java properties set on the **General** tab and with several pass-through values set on the **Pass Throughs** tab.

To create a record adapter to read from one or more Record Store instances:

1. Open your project in Developer Studio.
2. In the Project tab, double-click **Pipeline Diagram**.
3. In the Pipeline Diagram editor, click **New**.
4. Select **Record** > **Adapter**.
5. In the **Name** text box, specify the name of this record adapter.
6. In the **Direction** frame, select **Input**.
7. From the **Format** list, choose **Custom Adapter**.
8. In the **Class** field of Java Properties, specify one of the following:

- To read from one Record Store instance, specify `com.endeca.itl.recordstore.forge.RecordStoreSource`.
- To read from multiple Record Store instances, specify `com.endeca.itl.recordstore.forge.MultipleRecordStoreSource`. This class instructs Forge to contact the Component Instance Manager, request a list of all available Record Store instances, and read from each.

9. In the **Classpath** field of Java Properties, specify the path to `<install path>/CAS/`*`version`*`/lib/recordstore-forge-adapter/recordstore-forge-adapter-3.0.0.jar`.

   Endeca recommends that you keep this JAR file in the `lib` directory because of the large number of dependencies on other JAR files in that location.
10. Select the **Pass Throughs** tab of the Record Adapter editor.
11. On the **Pass Throughs** tab, create the following name/value pairs:

- Set a `HOST` pass-through to the fully qualified host name of the machine running the Endeca CAS Service. For example, `HOST = hostname.endeca.com`.
- Set a `PORT` pass-through to the port number that the Endeca CAS Service is listening on. For example, `PORT = 8500`.
- If reading from one Record Store instance, set an `INSTANCE_NAME` pass-through to the name of the Record Store instance that you want Forge to read from. For example, `INSTANCE_NAME = crawlID`. This pass-through is not required if the adapter is reading from multiple Record Store instances.
- For a baseline pipeline, set a `READ_TYPE` pass-through to `BASELINE`. The `BASELINE` setting instructs Forge to read the latest version of all records in the Record Store. For example, `READ_TYPE = BASELINE`.

  For a partial-update pipeline, set a `READ_TYPE` pass-through to `DELTA`. The `DELTA` setting instructs Forge to read records that have been modified or added between the last committed generation in the Record Store and the last generation read by the same client as identified by `CLIENT_ID` setting. For example, `READ_TYPE = DELTA`.
- Set a `CLIENT_ID` pass-through to a string that distinguishes this client from others that may also be reading from the Record Store instances. For example, `CLIENT_ID = FORGE`. The `CLIENT_ID` pass-through specifies the client ID to be set for the generation that is being read in. In effect, this pass-through is performing the set-last-read-generation task that can be performed with the CAS Server Command-line Utility (i.e., state is being set for the client, which is Forge in this case). This pass-through can be used only for `READ_TYPE` operations.
- Optionally, set a `RECORDS_PER_TRANSFER` pass-through to the number of records to transfer at a time for each Record Store instance. The default is `500`. Click **OK** to add the new record adapter to the project.
- Optionally, to enable SSL with server only authentication, add pass through options for the truststore location (`SSL_TRUSTSTORE`), type (`SSL_TRUSTSTORE_TYPE`), password (`SSL_TRUSTSTORE_PASSWORD`), and CAS port usage (`IS_PORT_SSL`).

  Note: A value of `true` means that `PORT` is an SSL port and the record adapter uses HTTPS for connections. A value of `false` means that `PORT` is a non-SSL port and the record adapter uses HTTPS redirects. Specify `false` if you enabled redirects from a non-SSL port to an SSL port.

  For example: `SSL_TRUSTSTORE = C:\Endeca\CAS\workspace\conf\truststore.ks`, `SSL_TRUSTSTORE_TYPE = JKS`, `SSL_TRUSTSTORE_PASSWORD = endeca`, `IS_PORT_SSL = false`.

- Optionally, to enable SSL with mutual authentication, add pass-through options for the keystore location (`SSL_KEYSTORE`), type (`SSL_KEYSTORE_TYPE`), and password (`SSL_KEYSTORE_PASS¬ WORD`).

  For example: `SSL_KEYSTORE = C:\Endeca\CAS\workspace\conf\keystore.ks`, `SSL_KEYSTORE_TYPE = JKS`, `SSL_KEYSTORE_PASSWORD = endeca`, `IS_PORT_SSL = false`.

12. Click **OK** to add the new record adapter to the project.
13. Save the project by selecting **Save** from the File menu.

In some cases, you may get an `Out of Memory` error if Forge is reading or writing records from a Record Store instance. To work around this error, you can increase the amount of memory allocated to the JVM running Forge. To increase the memory, run Forge with `--javaArgument` flag and the `-Xmx` argument, for example `--javaArgument -Xmx512m`.

Automating record processing

You can automate record processing asynchronously or synchronously.

Asynchronous processing

In this model, you can use Windows Scheduler or a UNIX cron job to schedule when the data sources are crawled. The job to crawl data sources and the job to run a Forge pipeline can overlap and run simultaneously.

As long as there is at least one generation of records in the Record Store, you can run a Forge pipeline to read and process that generation of records, while at the same time, other crawling processes write new generations of records to the Record Store.

Synchronous processing

In this model, you use a Deployment Template script to sequentially crawl the data sources, wait for the CAS crawl script to complete, and then run a baseline script to process the resulting records in a Forge pipeline.

What's next

For more detailed information about configuring and using CAS, see the *Endeca CAS Developer's Guide*. To build or modify a Forge pipeline, see the *Endeca Developer Studio Help*. For general feature development, see the *Endeca Basic Development Guide* and the *Endeca Advanced Development Guide*.