Oracle® Endeca Information Discovery Integrator

Integrator Acquisition System Developer's Guide

Version 3.0.0 • May 2013

ORACLE

Copyright and disclaimer

Table of Contents

Part III: IAS Command Line Utilities

Appendix B: Record Properties Generated by Crawling

Preface

Oracle® Endeca Information Discovery Integrator provides a suite of products to load data from disparate source systems and store it for use in an Endeca Server data domain. The Integrator products include:

- Integrator ETL - Integrator ETL is a high-performance data integration platform that lets you extract source records from a variety of sources and sends that data to the Data Ingest Web Service, which in turn loads the records into the Oracle Endeca Server.
- Integrator Acquisition System - The Integrator Acquisition System, or IAS, is a set of components that crawl source data stored in a variety of formats including: file systems, delimited files, JDBC databases, and custom data sources. IAS transforms the data, if necessary, and outputs the data to an XML file or a Record Store instance that can be accessed by Integrator ETL for use in the Endeca Server.
- IKM SQL to Endeca Server - provides integration and loading modules that enable writing source data to an Endeca Server target within Oracle Data Integrator.

About this guide

This guide describes how to configure and run IAS crawls to acquire data stored in file systems, Web servers, CMS repositories, and custom data sources. After crawling a data source, Endeca records are stored in a Record Store and become available for use in Integrator.

The guide assumes that you are familiar with Endeca concepts and Endeca application development.

Who should use this guide

This guide is intended for data developers who are using IAS to crawl data sources, manipulate the records if necessary, and incorporate the records into Integrator.

Conventions used in this guide

The following conventions are used in this document.

Typographic conventions

The following table describes the typographic conventions used in this document.

Table 0.1: Typographic conventions

Typeface	Meaning
User Interface Elements	This formatting is used for graphical user interface elements such as pages, dialog boxes, buttons, and fields.
`Code Sample`	This formatting is used for sample code phrases within a paragraph.

Typeface	Meaning
`<Variable Name>`	This formatting is used for variable values, such as <install path>.
`File Path`	This formatting is used for file names and paths.

Symbol conventions

The following table describes symbol conventions used in this document.

Table 0.2: Symbol conventions

Symbol	Description	Example	Meaning
>	The right angle bracket, or greater-than sign, indicates menu item selections in a graphic user interface.	File > New > Project	From the File menu, choose New, then from the New submenu, choose Project.

Contacting Oracle Customer Support

Oracle Customer Support provides registered users with important information regarding Oracle software, implementation questions, product and solution help, as well as overall news and updates from Oracle.

You can contact Oracle Customer Support through Oracle's Support portal, My Oracle Support at *https://support.oracle.com*.

Part I

Introduction to IAS

Chapter 1

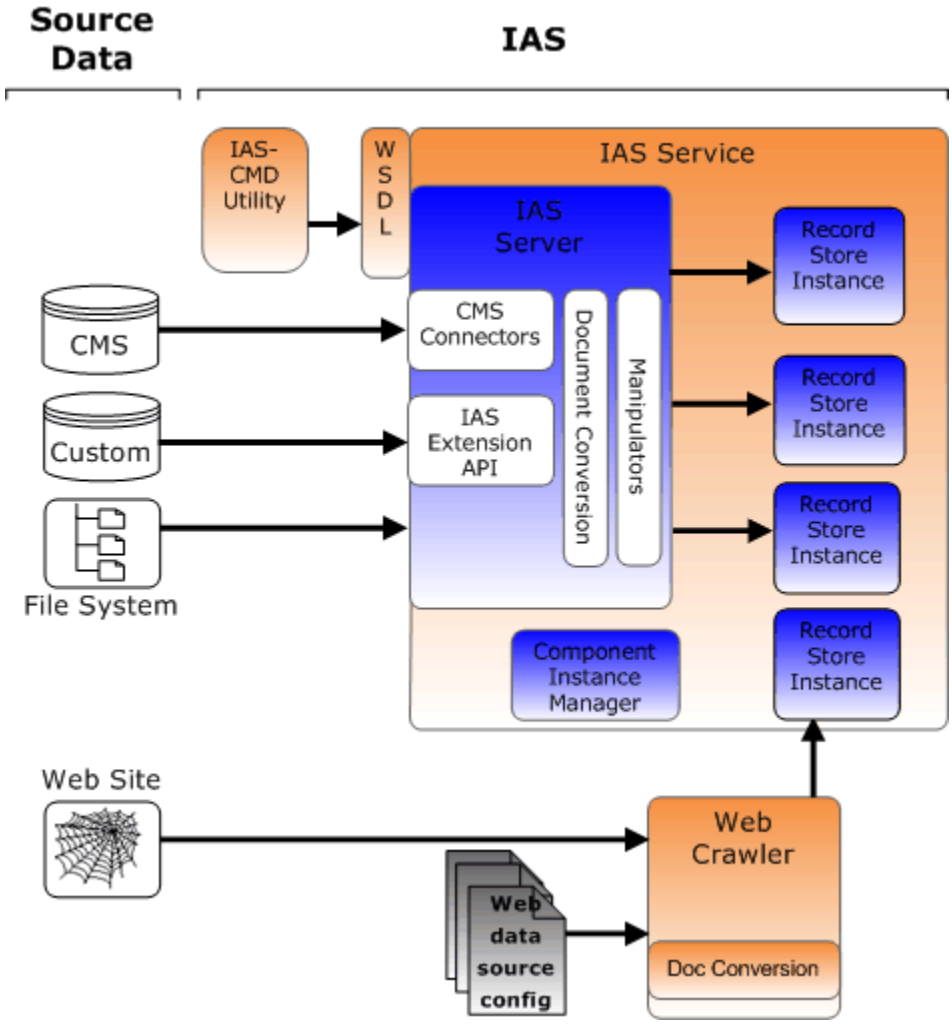
Introduction

This section provides introductory information about the Endeca Integrator Acquisition System (IAS).

Overview of the Integrator Acquisition System

The Integrator Acquisition System is a set of components that crawl data sources and produce Endeca records for use in an Endeca application. Data sources include file systems, content management systems, Web servers, and custom data sources.

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



IAS Components

The Integrator Acquisition System is made up of the following components:

- The Endeca IAS Service is a servlet container that runs the IAS Server, the Component Instance Manager, and any number of Record Store instances (one per crawl).
- The IAS Server is the component that manages all crawling operations.

- The IAS Server API allows users to write programs that communicate with the IAS Server. The IAS Server API has a WSDL interface and also a IAS Server Command-line Utility. The API is documented in the *IAS API Guide*.
- The Endeca Web Crawler manages all Web crawl-related operations. This component is documented in the *IAS Web Crawler Guide*.
- Endeca crawls provide a means to access data sources in a wide variety of CMS types, such as Documentum and Microsoft SharePoint.
- 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 IAS Server writes crawl output from each crawl to a unique Record Store instance.
- The IAS 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 Integrator Acquisition System. After you install the plug-in, the extensions are available and configurable using the IAS Server API and the IAS Server Command-line Utility. This API is documented in the *Integrator Acquisition System Extension API Guide*.

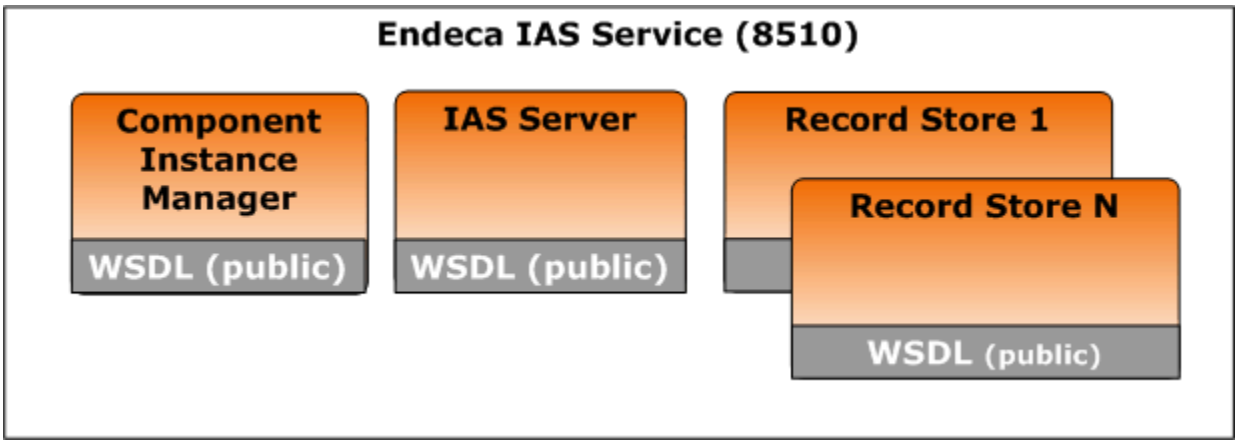
Interaction with Integrator

After running a crawl, IAS stores the resulting Endeca records in a Record Store. The records are then available for use in Integrator. In a typical data processing scenario, you create a Record Store Reader Component in Integrator and configure the component to connect to the Record Store. The Record Store Reader Component reads the records and an Integrator graph processes them as necessary. For details about the Record Store Reader Component, see the *Integrator User's Guide* available on the Oracle Technology Network.

About the Endeca IAS Service

The Endeca IAS Service is a servlet container that runs the IAS Server, the Component Instance Manager, and any number of Record Store instances (one per crawl).

On Windows, the IAS installation program starts the service automatically and the service is set to restart automatically during system restarts. If you accept the installation defaults, the service runs on port 8510. The following image shows the components running within the Endeca IAS Service:



In the Windows Services console, the service displays as Endeca IAS Service. The service is running `ias-service-wrapper.exe` in `<install path>\IAS\<version>\bin`.

On UNIX, you can start Endeca IAS Service using `ias-service.sh` located in `<install path>/IAS/<version>/bin` and stop it using `ias-service-shutdown.sh`. Or you can set up the service in `inittab`. For details, see the *Integrator Acquisition System Installation Guide*.

About the IAS Server

The IAS Server manages all crawl operations of file system, CMS, and custom data sources. The IAS Server has a WSDL interface and an IAS Server Command Line Utility. The IAS Server runs inside the IAS Service.

The IAS Server has the following characteristics:

- Includes the IAS Document Conversion Module, which allows the IAS Server to convert binary files (such as Microsoft Word documents and Adobe Acrobat PDF files) into text.
- Uses include and exclude filters to specify which files or folders to retrieve or avoid.
- Configures the logging behavior for a crawl, including setting the log level for various components and specifying output to the console, to a log file, or to both.
- Supports incremental crawls in which the IAS Server processes only the content that has been added, modified, or deleted since the last crawl.
- Enables security by supporting the Endeca Access Control System. Each crawl generates access control list (ACL) properties for each record, based on the corresponding properties for each file (for file system crawls) or entry in the CMS repository (for CMS crawls).

The IAS Server also tags the records with metadata properties that are derived from the source documents. After the IAS Server returns the records, you can configure an Endeca Record Store Reader (in Integrator) to

read the records into your Endeca graph, where Integrator processes the records and can add or modify the record properties. You can then build an Endeca application to access the records and allow your application users to search and navigate the document contents contained in the records.

You can configure the IAS Server to generate access control list (ACL) properties for each record that a crawl produces. These ACL properties can be used in conjunction with security login modules to limit access to records based on user login profiles. For details on using the Endeca Access Control System, see the *Endeca Information Discovery Integrator Security Guide*.

About the Component Instance Manager

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 Component Instance Manager runs inside the Endeca IAS Service.

About the Record Store

The Endeca Record Store is a Web service that provides persistent storage for generations of records that can later be accessed by Integrator for use in a graph. The Endeca Record Store is integrated with the Endeca IAS Server to directly store output in the Record Store instead of sending output to files.

The Record Store has the following features:

Provides an efficient repository for records

Instead of storing source records in different directories, they can be consolidated in one place. This consolidation eliminates the need to copy and move source files among different directories.

Operates asynchronously

The IAS Server can write records into a Record Store while, at the same time, Integrator can read records to process in a graph. Each process is isolated from changes that the other is currently making.

Creates a separate Record Store instance for each crawl

The Record Store Web service creates a unique Record Store instance for each crawl that IAS Server runs. In general, there is a one-to-one mapping from a crawl to a corresponding Record Store instance. A separate Record Store instance for each data source keeps record schemas separate. IAS enforces this one-to-one mapping by creating a new Record Store instance for each crawl you create. This mapping is not enforced in cases where you explicitly disable auto management using the `isManaged` property.

Automatically cleans stale records

The Record Store service periodically removes stale generations of records. The time interval to remove stale generations is configurable and the feature can be disabled if necessary.

Easily configured and managed with a Record Store Command-line utility

IAS includes a Record Store Command-line utility to perform Record Store configuration and management. You can use this utility to run get/set commands to update a Record Store instance with configuration settings.

You create a Record Store instance using either the Component Instance Manager Command-line Utility or the IAS Server API.

About using SSL with IAS

Configuring SSL in the Integrator Acquisition System enables SSL communication among all the IAS components. For information about how to configure SSL in IAS, see the *Security Guide for Integrator*.

Overview of the default IAS crawls and manipulators

The Integrator Acquisition System ships with a set of default crawls and manipulators. Each is briefly described here:

Crawl Type	Description
Delimited File	Crawls records in delimited text files, including .csv files.
Documentum Content Server	Crawls Documentum Content Server repositories (docbases).
File System	Crawls folders and files on both local drives and network drives.
JDBC	Crawls a JDBC-accessible database.
Microsoft SharePoint Object Model	Crawls SharePoint repositories using a custom Web Service that is implemented with the SharePoint Object Model API.
Microsoft SharePoint Web Services	Crawls SharePoint repositories using the default SharePoint Web Services interface.

Manipulator	Description
Filtering Script	This manipulator runs an inline BeanShell script that filters Endeca records from crawl output.
Modifying Script	This manipulator runs an inline BeanShell script that modifies Endeca records.

For information about configuring a crawl or a manipulator, run the `ias-cmd` utility with the `getModuleSpec` task to return configuration properties.

Chapter 2

Running the IAS Sample Applications

This section describes the sample applications.

About the sample IAS applications

This section describes how to run the sample applications to exercise the functionality of IAS.

There are two sample applications:

- A Java client that communicates with a Record Store instance and issues record access requests. This is stored in `IAS\<version>\sample\recordstore-java-client`.
- A Java client that communicates with the IAS Service and issues file system crawling requests. This is stored in `IAS\<version>\sample\ias-server-java-client`.

Using the IAS Server Java client

The Endeca IAS Server API allows users to build client programs that invoke the Endeca IAS Server to programmatically modify and control a variety of file system and CMS crawling operations.

IAS Server Java client sample files and directories

This topic describes the contents of the IAS Server Java Client directory.

The IAS Server Java Client (in the `/sample` directory) has the following directory structure:

```
/ias-server-java-client
  /lib
  /src
  .classpath
  .project
  build.xml
```

The contents are as follows:

- `lib` – Contains the Java libraries for the IAS Server Java client application.
- `src` – Contains the Java source file for the IAS Server Java Client application.
- `.classpath` – The classpath file for the Eclipse project.

- `.project` – The Eclipse project file for the recordstore-java-client project.
- `build.xml` – The Ant build file for the Record Store Java client application.

About the IAS Server Java client program

The IAS Server Java Client (as coded in the `IasServerSampleClient.java` source file) demonstrates a number of basic crawling operations.

The Endeca IAS Server Java Client is intended to provide a working example of a client that communicates with a running IAS Server and issues file system crawling requests. The sample client program is therefore a template that you can use as a basis for your own client program.

The package includes all the libraries needed to build clients. It also includes an Ant build script (which can compile and run the sample program) as well as Eclipse `.project` and `.classpath` files for the sample client.

Important: Please note that before starting Eclipse, you should run at least `ant compile` so that Eclipse can find the generated Web service stubs.

The sample client application performs the following actions:

1. Makes a connection to the IAS Service.
2. Creates a new file system crawl (named SampleClientTestCrawl), with the current working directory of the sample client (`.\` on Windows or `./` on UNIX) as the seed.
3. Runs a full crawl.
4. Updates the crawl configuration by adding file filters and enabling document conversion.
5. Runs a second full crawl, this time using the new filters and extracting text from documents.
6. Deletes the sample crawl.

Note that a default time limit of 10 seconds is set on both crawls, which means that in most cases the crawl output will not contain all the files on your file system.

The output files are written to the `workspace/output/SampleClientTestCrawl` directory, using a non-compressed XML file format. You can use a text editor to view the contents of the output.

Building and running the Java client with Ant

The Ant `build.xml` file can compile and run the sample client program.

As with any Ant build file, you can run a specific target or run them all at once. Before starting Eclipse, you should run at least the `compile` target so that Eclipse can find the generated Web service stubs.

The file has the following targets:

- `compile` - Runs javac to compile the generated client stubs and sample application.
- `run-demo` - Runs the previous two targets and then runs the sample client application.
- `clean` - Deletes the build directory.

To run the Ant build script:

1. Start the Endeca IAS Service if it is not already running.
   - Windows: Start the IAS Service from the Windows Services console.
   - UNIX: Run the `ias-service.sh` script.
2. From a command prompt, navigate to the `ias-server-java-client` directory and issue the following command to compile and run the sample client demo:
   `ant run-demo [--host <host name>] [--port <port number>]`

Note: You can issue the `ant compile` command if you just want to compile (but not run) the sample client program.

The demo file system crawl (named `SampleClientTestCrawl`) will use `C:\` on Windows and `/` on UNIX as the seed. When the demo crawl finishes, the IAS Service's `workspace/output/SampleClientTestCrawl` directory should contain two XML-format output files: `CrawlerOutput-FULL.xml` will have the content of the second crawl (i.e., the updated crawl with file filters), while the time-stamped file in the `archive` directory will have the content from the first crawl.

Opening the ias-server-java-client project in Eclipse

If you use Eclipse for your projects, the sample client package includes Eclipse `.project` and `.classpath` files.

To load the sample client project:

1. Make sure that you have run the Ant build file with at least the `compile` target. This generates the necessary Web service stubs.
2. Start Eclipse.
3. Import the project:
   (a) Open the **File** menu.
   (b) Click **Import....**
   (c) Expand the **General** folder.
   (d) Select **Existing Projects into Workspace**.
   (e) Select the `ias-server-java-client` project.
   (f) Click **Finish**.

Running the operations of the Java client

You should note certain important operations of the `Main` class in the `IasServerSampleClient.java` source.

1. The values for the host and port of the IAS Service are set by first reading the `workspace\conf\commandline.properties` file. If they do not exist, defaults of `localhost` and `8510` are used.

```
String host = System.getProperty(IAS_HOST_PROPERTY);
        String port = System.getProperty(IAS_PORT_PROPERTY);

        if (host == null || "".equals(host)) {
            host = EidiConstants.DEFAULT_HOST;
```

```
        }
        if (port == null || "".equals(port)) {
            port = EidiConstants.DEFAULT_PORT+"";
        }
```

2. Using the WSDL URL values, create a Web service locator and then use the `IasCrawlerLocator.getService()` method to get a handle to the IAS Service port.

```
IasCrawlerLocator locator = IasCrawlerLocator.create(host, Integer.parseInt(port));
        IasCrawler crawler = locator.getService();
```

3. Using a `CrawlId` object, set the name of the crawl in the constructor.

```
CrawlId crawlId = new CrawlId("SampleClientTestCrawl");
```

4. Using the `sampleCreateCrawl` method, create the new file system crawl. Text extraction is not enabled, which means that a probe crawl will be run. Note that the `IasCrawler.createCrawl()` method actually creates the crawl.

```
System.out.println("Creating Crawl with CrawlId '" + crawlId.getId() + "' ...");
sampleCreateCrawl(crawler, crawlId);
```

5. Using the `sampleRunFullCrawl` method, run the probe crawl, specifying a maximum of 10 seconds for the crawl duration. The `IasCrawler.startCrawl()` method is used to actually start the crawl, and then the `IasCrawler.stopCrawl()` method is used to stop the crawl after 10 seconds has elapsed.

```
System.out.println("Running probe crawl...");
sampleRunFullCrawl(crawler, crawlId, 10);
```

6. Using the `sampleUpdateCrawlAddingFiltersAndTextExtraction` method, enable text extraction and set wildcard (`htm*`) filters that are evaluated against the `Endeca.FileSystem.Extension` record property. The original crawl configuration is retrieved with the `IasCrawler.getCrawlConfig()` method and the updated configuration is sent to the IAS Server with the `IasCrawler.updateConfig()` method.

```
System.out.println("Adding filters and enabling text extraction...");
sampleUpdateCrawlAddingFiltersAndTextExtraction(crawler, crawlId);
```

7. Using the `sampleRunFullCrawl` method, run a second full crawl that does text extraction and uses the added filters. As with the previous crawl, a maximum of 10 seconds is specified for the crawl duration.

```
System.out.println("Running full crawl...");
sampleRunFullCrawl(crawler, crawlId, 10);
```

8. Using the `sampleDeleteCrawl` method, delete the SampleClientTestCrawl demo crawl. Note that the class uses the `IasCrawler.deleteCrawl()` method to actually delete the crawl.

```
System.out.println("Deleting crawl...");
sampleDeleteCrawl(crawler, crawlId);
```

The sample client program also shows the use of other IAS Server API functions, such as the `IasCrawler.listCrawls()`, `IasCrawler.getStatus()` and `IasCrawler.getMetrics()` methods.

You can modify the file and add other crawling operations, such as changing the output options (to send output to a Record Store instance), adding other types of filters (including date and regex filters), enabling archive expansion, and even returning information about the IAS Server. You can also use the sample code as a basis for creating and running CMS crawls.

Using the Record Store Java client

The Endeca Record Store Java Client package is intended to provide a working example of a client that communicates with a Record Store instance and issues record access requests. The sample client program is therefore a template that you can use as a basis for your own client program.

The Endeca Record Store API allows users to build client programs that invoke an Endeca Record Store instance to programmatically write records to and read records from the Record Store.

The Record Store API consists of two components:

- Record Store core (WSDL) classes. These are classes that you generate from the Record Store WSDL file using a third-party tool (such as Apache CXF 2.0). For the sake of convenience, Java versions of these classes are included in the `recordstore-api-3.0.0.jar` library in the sample client package.
- Record Store utility (helper) classes, such as the `RecordStoreLocator`, `RecordStoreReader`, and `RecordStoreWriter` classes, which are used in the sample client applications. These Java classes are also included in the `recordstore-api-3.0.0.jar` library.

The sample client package includes all the libraries needed to build clients. It also includes an Ant build script (which can compile and run the sample applications) as well as Eclipse `.project` and `.classpath` files for the sample client.

For details about the Record Store API, see the *Integrator Acquisition System API Guide*.

Record Store client sample files and directories

This topic describes the contents of the Record Store Java Client directory.

The Record Store Java Client has the following directory structure:

```
/recordstore-java-client
  /conf
  /lib
  /src
  .classpath
  .project
  build.xml
  run-sample-reader.bat
  run-sample-reader.sh
  run-sample-writer.bat
  run-sample-writer.sh
```

The contents are as follows:

- `conf` – Contains the `log4j.properties` logger configuration file for the sample client application.
- `lib` – Contains the Java libraries for the Record Store Java client application.
- `src` – Contains the Java source files for the Record Store java client application.
- `.classpath` – The classpath file for the Eclipse project.
- `.project` – The Eclipse project file for the recordstore-java-client project.
- `build.xml` – The Ant build file for the Record Store Java client application.
- The scripts to run the sample reader and sample writer client applications (`run-sample-reader.sh` and `run-sample-writer.sh` for UNIX, and `run-sample-reader.bat` and `run-sample-writer.bat` for Windows).

About the Record Store sample client applications

The two Record Store sample client applications demonstrate the write and read functionality of the Record Store API.

The writer client

The writer client (in the `SampleWriter.java` source file) performs the following actions:

1. Creates a record that will be written to the Record Store.
2. Makes a connection to a Record Store instance, assumed to reside on the `localhost` machine with a port of `8510`.
3. Starts a `READ_WRITE` transaction.
4. Using the `RecordStoreWriter` methods, writes the record to the Record Store.
5. Commits the write transaction.

The reader client

The reader client (in the `SampleReader.java` source file) performs the following actions:

1. Makes a connection to a Record Store instance, assumed to reside on the `localhost` machine with a port of `8510`.
2. Starts a READ transaction.
3. Gets the ID of the last-committed generation.
4. Using the `RecordStoreReader.next()` method, reads the record from the Record Store and then writes its contents to standard output.
5. Commits the read transaction.

Note: If either application throws a `RecordStoreFault` exception, it is caught and the transaction is rolled back.

Building and running the sample writer client with Ant

The Ant `build.xml` file can compile and run the sample writer client program.

The file has the following targets:

- `init` – Creates the build directory structure that will be used by the compile target.
- `compile` – Runs javac to compile the sample client application.
- `run-sample-writer` – Runs the previous two targets and then runs the sample client writer application.
- `run-sample-reader` – Runs the init and compile targets, and then runs the sample client reader application.
- `clean` – Deletes the build directory.

To run the sample writer client with the Ant build script:

1. Start the Endeca IAS Service if it is not already running.
   - Windows: Start the IAS Service from the Windows Services console.
   - UNIX: Run the `ias-service.sh` script.
2. From a command prompt, navigate to the `IAS\<version>\sample\recordstore-java-client` directory and issue the following command to compile and run the sample writer client demo:
   ```
   run-sample-writer
   ```

The sample writer client's output messages should be similar to this example:

```
C:\Oracle\Endeca\IAS\3.0.0\sample\recordstore-java-client>run-sample-writer.bat

C:\Oracle\Endeca\IAS\3.0.0\sample\recordstore-java-client>REM Sample assumes ias
 service is running on http://localhost:8510/
Buildfile: C:\Oracle\Endeca\IAS\3.0.0\sample\recordstore-java-client\build.xml

init:

compile:
    [javac] C:\Oracle\Endeca\IAS\3.0.0\sample\recordstore-java-client\build.xml:
17: warning: 'includeantruntime' was not set, defaulting to build.sysclasspath=l
ast; set to false for repeatable builds

run-sample-writer:
     [java] Starting a new transaction ...
     [java] Writing records ...
     [java] Committing transaction ...
     [java] DONE

BUILD SUCCESSFUL
Total time: 5 seconds
```

You can use the `-c` (count) option with the read-baseline task of the Record Store Command-line Utility to determine if the Record Store has any records:

```
C:\Oracle\Endeca\IAS\3.0.0\bin> recordstore-cmd.bat read-baseline -a rs1 -c
Records read: 2
```

Building and running the sample reader client with Ant

The Ant `build.xml` file can compile and run the sample reader client program.

The file has the following targets:

- `init` – Creates the build directory structure that will be used by the compile target.
- `compile` – Runs javac to compile the sample client application.
- `run-sample-writer` – Runs the previous two targets and then runs the sample client writer application.
- `run-sample-reader` – Runs the init and compile targets, and then runs the sample client reader application.
- `clean` – Deletes the build directory.

To run the sample reader client with the Ant build script:

1. Start the Endeca IAS Service if it is not already running.
   - Windows: Start the IAS Service from the Windows Services console.

- UNIX: Run the `ias-service.sh` script.

2. From a command prompt, navigate to the `recordstore-java-client` directory and issue the following command to compile and run the sample reader client demo:

```
run-sample-reader
```

The sample reader client's output messages should be similar to this example:

```
C:\Oracle\Endeca\IAS\3.0.0\sample\recordstore-java-client>REM Sample assumes ias
 service is running on http://localhost:8510/
Buildfile: C:\Oracle\Endeca\IAS\3.0.0\sample\recordstore-java-client\build.xml

init:

compile:
    [javac] C:\Oracle\Endeca\IAS\3.0.0\sample\recordstore-java-client\build.xml:
17: warning: 'includeantruntime' was not set, defaulting to build.sysclasspath=l
ast; set to false for repeatable builds

run-sample-reader:
     [java] Starting a new transaction ...
     [java] Getting the last committed generation ...
     [java] Reading records ...
     [java]   RECORD: [Endeca.Id=record1, fruit=apple, color=red]
     [java]   RECORD: [Endeca.Id=record2, fruit=banana, color=yellow]
     [java] 2 record(s) read
     [java] Committing transaction ...
     [java] DONE

BUILD SUCCESSFUL
Total time: 8 seconds
```

As the example shows, 2 records are read.

Opening the recordstore-java-client project in Eclipse

If you use Eclipse for your projects, the sample client package includes Eclipse `.project` and `.classpath` files.

As a prerequisite, make sure that you have run the Ant build file with at least the `compile` target. This will generate the necessary Web service stubs.

To load the sample client project:

1. Start Eclipse.
2. Import the project:
   (a) Open the **File** menu.
   (b) Click **Import...**
   (c) Expand the **General** folder.
   (d) Select **Existing Projects into Workspace**
   (e) Select the `recordstore-java-client` project.
   (f) Click **Finish**.

Running the operations of the sample writer client

This section provides an overview of the more important operations of the sample writer client program. You can modify the files and add other Record Store operations.

The methods for these operations are described in the *IAS API Guide* and in the *Record Store API Reference (Javadoc)*.

The `SampleWriter.java` source program executes the following important operations:

1. A constant is set for the value of the `idPropertyName` configuration property that is used for the Record Store instance.
   ```
   public static final String PROPERTY_ID = "Endeca.Id";
   ```
2. Using the `RecordStoreLocator` utility class, create a Web service locator with host name, port number, and Record Store instance name:
   ```
   RecordStoreLocator locator = RecordStoreLocator.create("localhost", 8510, "rs1");
   ```
3. Use the `RecordStore.getService()` method to establish a connection to the Record Store instance:
   ```
   RecordStore recordStore = locator.getService();
   ```
4. Using the transaction ID created by the `RecordStore.startTransaction()` method, the `RecordStoreWriter.createWriter()` method is used to create a writer.
   ```
   transactionId = recordStore.startTransaction(TransactionType.READ_WRITE);

   RecordStoreWriter writer =
      RecordStoreWriter.createWriter(recordStore, transactionId);
   ```
5. The writer first writes a "Delete All" record, then writes the sample record, and finally closes the writer.
   ```
   writer.deleteAll();
   writer.write(records);
   writer.close();
   ```
6. The `RecordStore.commitTransaction()` method closes the transaction.
   ```
   recordStore.commitTransaction(tId);
   ```

Note: If a `RecordStoreFault` exception occurs during the write process, it is caught and the `RecordStore.rollbackTransaction()` method rolls back the `READ_WRITE` transaction.

Running the operations of the sample reader client

This section provides an overview of the more important operations of the sample reader client program. You can modify the files and add other Record Store operations.

The methods for these operations are described in the*IAS API Guide* and in the *Record Store API Reference (Javadoc)*.

The `SampleReader.java` source program executes the following important operations:

1. Using the `RecordStoreLocator` utility class, create a Web service locator with host name, port number, and Record Store instance name:

```
RecordStoreLocator locator
= RecordStoreLocator.create("localhost", 8510, recordStoreInstance);
```

2. Use the `RecordStore.getService()` method to establish a connection to the Record Store instance:

```
RecordStore recordStore = locator.getService();
```

3. Using the transaction ID created by the `RecordStore.startTransaction()` method, the `RecordStore.getLastCommittedGenerationId()` method is used to get the ID of the last-committed generation in the Record Store.

```
transactionId = recordStore.startTransaction(TransactionType.READ);

GenerationId gId = recordStore.getLastCommittedGenerationId(transactionId);
```

4. The `RecordStoreReader.createBaselineReader()` method uses the transaction ID and the generation ID to create a reader.

```
RecordStoreReader reader =
   RecordStoreReader.createBaselineReader(recordStore, transactionId, gId);
```

5. Within a while loop, the `RecordStoreReader` `hasNext()` and `next()` methods are used to read the sample record. The reader is closed when there are no more records to be read.

```
int count = 0;
while (reader.hasNext()) {
   Record record = reader.next();
   System.out.println("  RECORD: " + record);
   count++;
}
reader.close();
```

6. The `RecordStore.commitTransaction()` method closes the transaction.

```
recordStore.commitTransaction(tId);
```

Note: As with the writer client, if a `RecordStoreFault` exception occurs during the read process, it is caught and the `RecordStore.rollbackTransaction()` method rolls back the READ transaction.

Part II

Crawling Data Sources

Chapter 3

Creating a Crawl

This section describes how to create a crawl using the IAS Server Command-line Utility (`ias-cmd`). You can also create crawls programmatically using the IAS Server API. However, that approach is documented in the *Integrator Acquisition System API Guide* and the *IAS Server API Reference (Javadoc)*.

About creating a crawl

You create a crawl by creating a crawl configuration file and passing the file as an argument (`-f`) to the `createCrawls` task of the `ias-cmd` utility. By default, all crawls are configured to write output to a Record Store instance. If necessary, you can reconfigure a crawl to write output to a file (either XML or binary).

For each crawl, you specify configuration options such as:

- The name of the crawl.
- The location of the source data. For example, this could be a seed for a file system crawl, or a CMS repository for a CMS crawl, or a database for a JDBC crawl.
- Filters that include or exclude designated files and folders.
- Repository properties for CMS data sources.

- Manipulators to modify Endeca records as part of the crawl.

The process to create a crawl is the same for any crawl type including JDBC, Delimited File, File System, Documentum Content Server, and so on.

The steps to create a crawl are as follows:

1. Make a copy of one of the default crawl configuration files provided in `<install path>\IAS\<version>\sample\crawlConfigFiles`. (These files provide a convenient starting point.)
2. In the new file, specify configuration settings as appropriate for your source data and environment.
3. Save and close the crawl configuration file.
4. Upload the crawl configuration file to IAS by running the `createCrawls` task of `ias-cmd` and specifying the `-f` option with a path to the crawl configuration file.

For example, this command creates a new crawl configured by the file `crawlConfig.xml`.

```
C:\Oracle\Endeca\IAS\3.0.0\bin>ias-cmd createCrawls -f C:\tmp\crawlConfig.xml
```

After you create a crawl, you can run it using the `startCrawl` task of `ias-cmd`.

Creating a Delimited File crawl

You create a Delimited File crawl by making a copy of the default crawl configuration file for a Delimited File crawl and modifying the settings in the file as appropriate for your environment.

To create a Delimited File crawl:

1. In your IAS installation, locate the default crawl configuration files provided in `<install path>\IAS\<version>\sample\crawlConfigFiles`.
2. Make a copy of `delimitedFileCrawl.xml`, rename the file for your environment, and save it to a local directory.
3. Open the new crawl configuration file in a text editor.
4. Configure the settings that are common to all crawl types:

Option	Description
`crawlId`	(Required). Specify a unique name to distinguish the crawl from others in IAS. The `crawlId` can contain alphanumeric characters, underscores, dashes, and periods. All other characters are invalid for a `crawlId`.

Option	Description
`unavailableIncrementalSwitchesToFullCrawl`	(Optional). Specify a Boolean value to indicate whether IAS should switch from running an incremental crawl to running a full crawl in cases where it is not possible to run an incremental crawl. A value of `true` instructs IAS to run a full crawl if it is not possible to run an incremental crawl. A value of `false` instructs IAS to abort the incremental crawl and throw a `FullCrawlRequiredException` indicating why the incremental crawl could not run. The default value depends on the `outputConfig` type. If the `outputConfig` is set to `Record Store`, then the default value is `true`. If the `outputConfig` is set to `File System`, then the default value is `false`.
`crawlThreads`	(Optional). Specify the maximum threads available to the IAS Service. The default number of threads is one more than the number of CPUs of the machine running the IAS Service. If you are running multiple crawls concurrently, Oracle does not recommend increasing the value of this setting because it is possible for the system to over thread and consequently slow IAS performance. You can minimize the risk of over threading by reducing the value of this setting to no more than the number of CPUs plus 1 for each crawl running on a single IAS system.
`textExtractionConfig`	(Optional). Specify whether document conversion is enabled. For a delimited file crawl, `textExtractionConfig` must be set to `false`.
`manipulatorConfigs`	(Optional). Specify any number of manipulators within a crawl configuration. If one or more `manipulatorConfig` elements are present, IAS passes each record to each manipulator for processing according to its `manipulatorConfig` settings. Manipulators execute in the order in which they are nested within `manipulatorConfigs`.

5. Configure the settings that are specific to a Delimited File crawl:

Option	Description
`moduleId`	(Required). Specify the name of the IAS module for a Delimited File crawl. This value must be set to `com.endeca.ias.source.DelimitedFile`.

`inputFilePath`	(Required). Specify an absolute path to the delimited files you want to crawl. Wildcards may be used in the filename but not in the path preceding the filename.
	Example of syntax for local folders on Windows:
	`C:\Oracle\Endeca\test\data\incoming\records.txt`
	Example of syntax for Windows network drives:
	`\\abchost.endeca.com\documents\customers.csv`

Option	Description
recordIdColumn	(Required). Specify the name of the column that you want to map to the record ID property in the generated records.
delimiterChar	(Required). Specify a single character that delimits the fields in the records. The default delimiter is a comma (,).
quoteCharacter	(Required). Specify a single character that escapes occurrences of the delimited character within a field. The default quote character is a quote (").

columnNames	(Optional). Specify column names in the order in which they appear in a delimited text file. This optional configuration is typically only necessary in cases where a delimited file does not contain a header row. If **Column Names** are unspecified, the crawl treats the first row of the file as the header row and uses the column names as Endeca property names. This property can be multi-valued in order to specify each column as a separate value. For example:
	`<moduleProperty>`
	`<key>columnNames</key>`
	`<value>num</value>`
	`<value>name</value>`
	`<value>age</value>`
	`<value>hobby</value>`
	`</moduleProperty>`

multiAssignDelimiterChar	(Optional). Specify a single character that delimits multi-assign values within a multi-assign column. If you specify a value and omit adding any **Multi-Assign Columns**, the crawl parses all columns in the file as if they may contain multi-assign values.

multiAssignColumns	(Optional). Specify each column in the file that contains multi-assign values. For example, the `hobby` column may contain multi-assign values if a person has multiple hobbies:
	`<moduleProperty>`
	`<key>multiAssignColumns</key>`
	`<value>hobby</value>`
	`</moduleProperty>`

trimWhitespace	(Optional). Specify **true** to trim the leading or trailing whitespace from the data stored in columns of the delimited file. The default value is **true**.
characterEncoding	(Optional). Specify the character encoding of the delimited file that is being crawled. If unspecified, the default value is **UTF-8**.

6. Configure the settings to write crawl output to a Record Store. Although you can configure a crawl to write output to a file, writing to a Record Store is recommended and also the default. (To write to a file, see *Writing crawl output to a file on page 63*.)

Option	Description
`moduleId`	(Required). Specify the output type for a crawl. Specify an `id` of `Record Store` if the crawl is writing to a Record Store or specify `File System` if the crawl is writing to a file.
`isPortSsl`	(Optional). Specify whether to use SSL when connecting to the Record Store instance. A value of `true` uses HTTPS and treats the `port` property as an SSL port. A value of `false` uses HTTP and treats `port` as a non-SSL port. Specify `false` if you enabled redirects from a non-SSL port to an SSL port.
`host`	(Required). Specify the fully qualified name of the host running the Record Store instance. The default value is `localhost`.
`port`	(Required). Specify the port of the Endeca IAS Service running the Record Store instance. The default value is `8510`.
`instanceName`	(Optional). Specify the Record Store instance name to write to. If unspecified, this defaults to the same value as the `crawlId`.
`isManaged`	(Optional). Specify whether the Record Store is managed. If you disable the `isManaged` property by setting it to `false`, a Record Store instance is not created when you create the crawl. The default value is `true`.

7. Save and close the crawl configuration file.
8. Run the `createCrawls` task of `ias-cmd` to upload the crawl configuration file to IAS. See *Creating crawls on page 92*.

Creating a Documentum Content Server crawl

You create a Documentum Content Server crawl by making a copy of the default crawl configuration file for a Documentum Content Server crawl and modifying the settings in the file as appropriate for your environment.

Before creating a Documentum Content Server crawl, you must set up IAS to access Documentum shared libraries. See *Setting up IAS for Documentum Content Server on page 35*.

To create a Documentum Content Server crawl:

1. In your IAS installation, locate the default crawl configuration files provided in `<install path>\IAS\<version>\sample\crawlConfigFiles`.
2. Make a copy of `dctmContentServerCrawl.xml`, rename the file for your environment, and save it to a local directory.
3. Open the new crawl configuration file in a text editor.

4. Configure the settings that are common to all crawl types:

Option	Description
`crawlId`	(Required). Specify a unique name to distinguish the crawl from others in IAS. The `crawlId` can contain alphanumeric characters, underscores, dashes, and periods. All other characters are invalid for a `crawlId`.
`unavailableIncrementalSwitchesToFullCrawl`	(Optional). Specify a Boolean value to indicate whether IAS should switch from running an incremental crawl to running a full crawl in cases where it is not possible to run an incremental crawl. A value of `true` instructs IAS to run a full crawl if it is not possible to run an incremental crawl. A value of `false` instructs IAS to abort the incremental crawl and throw a `FullCrawlRequiredException` indicating why the incremental crawl could not run. The default value depends on the `outputConfig` type. If the `outputConfig` is set to `Record Store`, then the default value is `true`. If the `outputConfig` is set to `File System`, then the default value is `false`.
`crawlThreads`	(Optional). Specify the maximum threads available to the IAS Service. The default number of threads is one more than the number of CPUs of the machine running the IAS Service. If you are running multiple crawls concurrently, Oracle does not recommend increasing the value of this setting because it is possible for the system to over thread and consequently slow IAS performance. You can minimize the risk of over threading by reducing the value of this setting to no more than the number of CPUs plus 1 for each crawl running on a single IAS system.
`textExtractionConfig`	(Optional). Specify whether document conversion is enabled. If `textExtractionConfig` has a value of `true`, then IAS performs document conversion and stores the converted text as a property on the Endeca record.
`manipulatorConfigs`	(Optional). Specify any number of manipulators within a crawl configuration. If one or more `manipulatorConfig` elements are present, IAS passes each record to each manipulator for processing according to its `manipulatorConfig` settings. Manipulators execute in the order in which they are nested within `manipulatorConfigs`.

5. Configure the settings that are specific to a Documentum Content Server crawl:

Option	Description
`moduleId`	(Required). Specify the name of the IAS module for a Documentum Content Server crawl. This value must be set to `Documentum Content Server`.
`docbase`	(Required). Specify the name of the Documentum repository.

Option	Description
`domain`	(Optional). Specify the domain of the user name.
`expandArchives`	(Optional.) Enabling this option creates a record for each archived entry and populates the record's properties.
`password`	(Required). Specify a password for the username.

`seeds`	(Optional). Specify an relative path to a Documentum site. This property can be multi-valued in order to specify each seed separately. If unspecified, the default seed is the root of the repository.
	For example:
	`<moduleProperty>`
	`<key>seeds</key>`
	`<value>/dctm65/test</value>`
	`</moduleProperty>`

`username`	(Required). Specify a valid username that has access to the Documentum repository.
`webtopUrl`	(Optional). Specify the base URL of the local Documentum StringWebtop installation. For example: `http://myhost:8080/webtop/`.

6. Configure the settings to write crawl output to a Record Store. Although you can configure a crawl to write output to a file, writing to a Record Store is recommended and also the default. (To write to a file, see *Writing crawl output to a file on page 63*.)

Option	Description
`moduleId`	(Required). Specify the output type for a crawl. Specify an `id` of `Record Store` if the crawl is writing to a Record Store or specify `File System` if the crawl is writing to a file.
`isPortSsl`	(Optional). Specify whether to use SSL when connecting to the Record Store instance. A value of `true` uses HTTPS and treats the `port` property as an SSL port. A value of `false` uses HTTP and treats `port` as a non-SSL port. Specify `false` if you enabled redirects from a non-SSL port to an SSL port.
`host`	(Required). Specify the fully qualified name of the host running the Record Store instance. The default value is `localhost`.
`port`	(Required). Specify the port of the Endeca IAS Service running the Record Store instance. The default value is `8510`.
`instanceName`	(Optional). Specify the Record Store instance name to write to. If unspecified, this defaults to the same value as the `crawlId`.

Option	Description
`isManaged`	(Optional). Specify whether the Record Store is managed. If you disable the `isManaged` property by setting it to `false`, a Record Store instance is not created when you create the crawl. The default value is `true`.

7. Save and close the crawl configuration file.
8. Run the `createCrawls` task of `ias-cmd` to upload the crawl configuration file to IAS. See *Creating crawls on page 92*.

Supported versions of Documentum Content Server

IAS supports the following versions of Documentum Content Server repositories:

- Documentum Content Server 5.3
- Documentum Content Server 6.0
- Documentum Content Server 6.5

Note: Documentum Content Server 5.2.5 and earlier is not supported.

Setting up IAS for Documentum Content Server

To crawl a Documentum Content Server repository (docbases), you must configure the machine running IAS to access Documentum shared libraries.

Note: The following procedure uses the Documentum Foundation Classes 6.x and assumes a Windows installation of IAS. These classes are compatible with Documentum 5.3 and later.

To set up the IAS Server for Documentum:

1. Install and configure the Documentum Foundation Classes 6.x libraries using the Documentum installer. Refer to the Documentum documentation.
2. Copy the following file from the Documentum Foundation Classes installation to the `IAS\<version>\lib\ias-server-plugins\entropysoft` directory (on Windows) or `IAS/<version>/lib/ias-server-plugins/entropysoft` directory (on UNIX):
   - `dfc.jar` including any JAR files that `dfc.jar` depends upon. (You can identify dependencies by opening the manifest file for `dfc.jar` and copying the files listed in the manifest.)

     Note: By default, these DFC files are located under `C:\Program Files\Documentum\Shared` (on Windows).

3. Copy `dfc.properties` from the Documentum Foundation Classes installation directory to `IAS\workspace\conf` (on Windows).

   Note: By default, this file is located under `C:\Program Files\Documentum\config` (on Windows).

4. Restart the Endeca IAS Service.

Now the IAS Server is set up to communicate with the Documentum repository.

Limitations of a Documentum Content Server crawl

Renditions

Renditions are currently not supported. Only the original content can be retrieved or updated by the crawl. The "renditions" document property contains the list of available rendition formats for a given document.

Permission mapping in a Documentum Content Server crawl

The following table shows the mapping between Documentum permissions and the resulting Endeca record properties that are produced.

Documentum permission	Endeca record properties
Browse	`Endeca.CMS.AllowReadProperties`
Read	`Endeca.CMS.AllowReadContent` `Endeca.CMS.AllowReadProperties`
Relate	`Endeca.CMS.AllowReadContent` `Endeca.CMS.AllowReadProperties`
Version	`Endeca.CMS.AllowReadContent` `Endeca.CMS.AllowReadProperties`
Write	`Endeca.CMS.AllowReadContent` `Endeca.CMS.AllowReadProperties`
Delete	`Endeca.CMS.AllowReadContent` `Endeca.CMS.AllowReadProperties`

Creating a File System crawl

You create a File System crawl by making a copy of the default crawl configuration file for a File System crawl and modifying the settings in the file as appropriate for your environment.

To create a File System crawl:

1. In your IAS installation, locate the default crawl configuration files provided in `<install path>\IAS\<version>\sample\crawlConfigFiles`.
2. Make a copy of `fileSystemCrawl.xml`, rename the file for your environment, and save it to a local directory.

3. Open the new crawl configuration file in a text editor.
4. Configure the settings that are common to all crawl types:

Option	Description
`crawlId`	(Required). Specify a unique name to distinguish the crawl from others in IAS. The `crawlId` can contain alphanumeric characters, underscores, dashes, and periods. All other characters are invalid for a `crawlId`.
`unavailableIncrementalSwitchesToFullCrawl`	(Optional). Specify a Boolean value to indicate whether IAS should switch from running an incremental crawl to running a full crawl in cases where it is not possible to run an incremental crawl. A value of `true` instructs IAS to run a full crawl if it is not possible to run an incremental crawl. A value of `false` instructs IAS to abort the incremental crawl and throw a `FullCrawlRequiredException` indicating why the incremental crawl could not run. The default value depends on the `outputConfig` type. If the `outputConfig` is set to `Record Store`, then the default value is `true`. If the `outputConfig` is set to `File System`, then the default value is `false`.
`crawlThreads`	(Optional). Specify the maximum threads available to the IAS Service. The default number of threads is one more than the number of CPUs of the machine running the IAS Service. If you are running multiple crawls concurrently, Oracle does not recommend increasing the value of this setting because it is possible for the system to over thread and consequently slow IAS performance. You can minimize the risk of over threading by reducing the value of this setting to no more than the number of CPUs plus 1 for each crawl running on a single IAS system.
`textExtractionConfig`	(Optional). Specify whether document conversion is enabled. If `textExtractionConfig` has a value of `true`, then IAS performs document conversion and stores the converted text as a property on the Endeca record.
`manipulatorConfigs`	(Optional). Specify any number of manipulators within a crawl configuration. If one or more `manipulatorConfig` elements are present, IAS passes each record to each manipulator for processing according to its `manipulatorConfig` settings. Manipulators execute in the order in which they are nested within `manipulatorConfigs`.

5. Configure the settings that are specific to a File System crawl:

Option	Description
`moduleId`	(Required). Specify the name of the IAS module for a File System crawl. This value must be set to `File System`.

Option	Description

seeds	(Required). Specify an absolute path to a folder you want to crawl. Seeds may be local folders or network drives. Note that for Windows, you should specify network drives by universal naming convention (UNC) syntax rather than by using the letter of a mapped drive. For UNIX, you can specify mounted or local drives using standard file path syntax. This property can be multi-valued in order to specify each seed separately. For example:
	`<moduleProperty>`
	`<key>seeds</key>`
	`<value>C:\doc\ias</value>`
	`<value>C:\doc\mdex</value>`
	`<value>C:\doc\studio</value>`
	`</moduleProperty>`

gatherNativeFileProperties	(Optional.) Specify a Boolean value to create ACL properties in the records.
expandArchives	(Optional.) Specify whether to create a record for each archived entry and populate the record's properties.

6. Configure the settings to write crawl output to a Record Store. Although you can configure a crawl to write output to a file, writing to a Record Store is recommended and also the default. (To write to a file, see *Writing crawl output to a file on page 63*.)

Option	Description
moduleId	(Required). Specify the output type for a crawl. Specify an `id` of `Record Store` if the crawl is writing to a Record Store or specify `File System` if the crawl is writing to a file.
isPortSsl	(Optional). Specify whether to use SSL when connecting to the Record Store instance. A value of `true` uses HTTPS and treats the `port` property as an SSL port. A value of `false` uses HTTP and treats `port` as a non-SSL port. Specify `false` if you enabled redirects from a non-SSL port to an SSL port.
host	(Required). Specify the fully qualified name of the host running the Record Store instance. The default value is `localhost`.
port	(Required). Specify the port of the Endeca IAS Service running the Record Store instance. The default value is `8510`.
instanceName	(Optional). Specify the Record Store instance name to write to. If unspecified, this defaults to the same value as the `crawlId`.

Option	Description
`isManaged`	(Optional). Specify whether the Record Store is managed. If you disable the `isManaged` property by setting it to `false`, a Record Store instance is not created when you create the crawl. The default value is `true`.

7. Save and close the crawl configuration file.
8. Run the `createCrawls` task of `ias-cmd` to upload the crawl configuration file to IAS. See *Creating crawls on page 92*.

Creating a JDBC crawl

You create a JDBC crawl by making a copy of the default crawl configuration file for a JDBC crawl and modifying the settings in the file as appropriate for your environment.

Before creating a JDBC crawl, you must install the JDBC driver for the database that you want to crawl into the IAS `ias-server-plugins` directory. For example, if you are crawling an Oracle database, you install the JDBC driver for Oracle. Contact your DBA to determine the JDBC driver that is compatible with the version of the database you are running. IAS by default uses Java 1.6 (64-bit version), so the JDBC driver used must be Java 6 and 64-bit compatible.

To create a JDBC crawl:

1. Install a JDBC driver into IAS:
   (a) Stop Endeca IAS Service.
   (b) Navigate to `<install path>\IAS\<version>\lib\ias-server-plugins\ias-jdbc-datasource`.
   (c) Copy the JAR file for the JDBC driver into the `ias-jdbc-datasource` directory.
   (d) Start Endeca IAS Service.
2. In your IAS installation, locate the default crawl configuration files provided in `<install path>\IAS\<version>\sample\crawlConfigFiles`.
3. Make a copy of `jdbcCrawl.xml`, rename the file for your environment, and save it to a local directory.
4. Open the new crawl configuration file in a text editor.
5. Configure the settings that are common to all crawl types:

Option	Description
`crawlId`	(Required). Specify a unique name to distinguish the crawl from others in IAS. The `crawlId` can contain alphanumeric characters, underscores, dashes, and periods. All other characters are invalid for a `crawlId`.

Option	Description
`unavailableIncrementalSwitchesToFullCrawl`	(Required). Specify a Boolean value to indicate whether IAS should switch from running an incremental crawl to running a full crawl in cases where it is not possible to run an incremental crawl. A value of `true` instructs IAS to run a full crawl if it is not possible to run an incremental crawl. A value of `false` instructs IAS to abort the incremental crawl and throw a `FullCrawlRequiredException` indicating why the incremental crawl could not run. The default value depends on the `outputConfig` type. If the `outputConfig` is set to `Record Store`, then the default value is `true`. If the `outputConfig` is set to `File System`, then the default value is `false`.
`crawlThreads`	(Required). Specify the maximum threads available to the IAS Service. The default number of threads is one more than the number of CPUs of the machine running the IAS Service. If you are running multiple crawls concurrently, Oracle does not recommend increasing the value of this setting because it is possible for the system to over thread and consequently slow IAS performance. You can minimize the risk of over threading by reducing the value of this setting to no more than the number of CPUs plus 1 for each crawl running on a single IAS system.
`textExtractionConfig`	(Optional). Specify whether document conversion is enabled. If `textExtractionConfig` has a value of `true`, then IAS performs document conversion and stores the converted text as a property on the Endeca record.
`manipulatorConfigs`	(Optional). Specify any number of manipulators within a crawl configuration. If one or more `manipulatorConfig` elements are present, IAS passes each record to each manipulator for processing according to its `manipulatorConfig` settings. Manipulators execute in the order in which they are nested within `manipulatorConfigs`.

6. Configure the settings that are specific to a JDBC crawl:

Option	Description
`moduleId`	(Required). Specify the name of the IAS module for a JDBC crawl. This value must be set to `com.endeca.ias.source.JDBC`.
`driver`	(Required). Specify the fully qualified Java class name for the JDBC driver.
`jdbcUrl`	(Required). Specify the connection string that includes, at a minimum, the database vendor, the host and port, and the database instance name. If desired, you can also specify the **Username** and **Password** as part of the connection string.

Option	Description

connectionProperties	(Optional). Specify any additional connection properties that your database may require. Specify properties in the format: `name=value`. For example:
	`<moduleProperty>`
	`<key>connectionProperties</key>`
	`<value>user=testuser</value>`
	`<value>password=testpassword</value>`
	`</moduleProperty>`

sql	(Required). Specify a SQL query to execute against the database. IAS does not impose a length limit on the SQL statement contained in this configuration property. For example:
	`<moduleProperty>`
	`<key>sql</key>`
	`<value>SELECT *`
	`FROM people`
	`WHERE age = 30</value>`
	`</moduleProperty>`

keyColumn	(Required). Specify the name of the column in the database that you want to map to the record ID property in the generated records.

7. Configure the settings to write crawl output to a Record Store. Although you can configure a crawl to write output to a file, writing to a Record Store is recommended and also the default. (To write to a file, see *Writing crawl output to a file on page 63*.)

Option	Description
moduleId	(Required). Specifies the output type for a crawl. Specify an `id` of `Record Store` if the crawl is writing to a Record Store or specify `File System` if the crawl is writing to a file.
isPortSsl	(Required). Specifies whether to use SSL when connecting to the Record Store instance. A value of `true` uses HTTPS and treats the `port` property as an SSL port. A value of `false` uses HTTP and treats `port` as a non-SSL port. Specify `false` if you enabled redirects from a non-SSL port to an SSL port.
host	(Required). Specifies the fully qualified name of the host running the Record Store instance. The default value is `localhost`.
port	(Required). Specifies the port of the Endeca IAS Service running the Record Store instance. The default value is `8510`.

Option	Description
`instanceName`	(Optional). Specifies the Record Store instance name to write to. If unspecified, this defaults to the same value as the `crawlId`.
`isManaged`	(Optional). Specifies whether the Record Store is managed. If you disable the `isManaged` property by setting it to `false`, a Record Store instance is not created when you create the crawl. The default value is `true`.

8. Save and close the crawl configuration file.
9. Run the `createCrawls` task of `ias-cmd` to upload the crawl configuration file to IAS. See *Creating crawls on page 92*.

Feature notes and known limitations of JDBC crawls

BLOBs and document conversion

If the document conversion option is enabled, any columns that contain binary data are processed by the IAS Document Conversion Module.

Record spec and Key Column values

The data type of **Key Column** property cannot be a `BLOB` (binary large object) or other type of binary object (`BINARY`, `VARBINARY`, `LONGVARBINARY`).

BLOBs

A JDBC crawl supports a maximum of one column containing BLOB data per crawl.

Unsupported drivers

A JDBC crawl does not support the JDBC-ODBC bridge driver (`sun.jdbc.odbc.JdbcOdbcDriver`).

Unsupported stored procedures

A JDBC crawl does not support stored procedures. The `sql` configuration property must contain static SQL; it cannot contain a stored procedure.

Unsupported data types

A JDBC crawl does not support the following data types:

- `NULL`
- `OTHER`
- `JAVA_OBJECT`
- `DISTINCT`
- `STRUCT`

- ARRAY
- REF

Determining which SharePoint crawl to use

Both SharePoint crawls allow IAS to communicate with a SharePoint repository. However, each crawl communicates with SharePoint using a different type of interface. The SharePoint Object Model crawl uses a custom Web Service that is implemented using the SharePoint Object Model API. The SharePoint Web Services crawl uses the default SharePoint Web Services interface.

Typically, the type of interface that an IAS crawl uses would be a transparent implementation detail. In this case, the interface type may have capability limitations that may affect your application.

The following list provides guidance to determine which crawl is more suited to your needs:

- If you need to crawl either SharePoint Portal Server 2003 (SPS 2003) or Windows SharePoint Services 2.0 (WSS 2.0), then you should use the SharePoint Web Services crawl. The SharePoint Object Model crawl does not support those versions of SharePoint.
- If you need to crawl document-level ACL properties, then you should use the SharePoint Object Model crawl. The SharePoint Web Services crawl does not support crawling document-level ACL properties; however, it does support crawling site-level ACL properties.
- The SharePoint Object Model crawl requires that the "SharePoint solution deployment" be installed on the server running SharePoint. If you cannot install additional software on that server, you may have to use the SharePoint Web Services crawl.
- The SharePoint Web Services crawl has better performance than the SharePoint Object Model crawl.

Creating a SharePoint Object Model crawl

You create a SharePoint Object Model crawl by making a copy of the default crawl configuration file for a SharePoint Object Model crawl and modifying the settings in the file as appropriate for your environment.

Before you can create a SharePoint Object Model crawl, you must install a SharePoint solution on the SharePoint server. See *Installing a SharePoint solution on the SharePoint server on page 47*.

To create a SharePoint Object Model crawl:

1. In your IAS installation, locate the default crawl configuration files provided in `<install path>\IAS\<version>\sample\crawlConfigFiles`.
2. Make a copy of `sharePointOMCrawl.xml`, rename the file for your environment, and save it to a local directory.
3. Open the new crawl configuration file in a text editor.

4. Configure the settings that are common to all crawl types:

Option	Description
`crawlId`	(Required). Specify a unique name to distinguish the crawl from others in IAS. The `crawlId` can contain alphanumeric characters, underscores, dashes, and periods. All other characters are invalid for a `crawlId`.
`unavailableIncrementalSwitchesToFullCrawl`	(Required). Specifies a Boolean value to indicate whether IAS should switch from running an incremental crawl to running a full crawl in cases where it is not possible to run an incremental crawl. A value of `true` instructs IAS to run a full crawl if it is not possible to run an incremental crawl. A value of `false` instructs IAS to abort the incremental crawl and throw a `FullCrawlRequiredException` indicating why the incremental crawl could not run. The default value depends on the `outputConfig` type. If the `outputConfig` is set to `Record Store`, then the default value is `true`. If the `outputConfig` is set to `File System`, then the default value is `false`.
`crawlThreads`	(Required). Specify the maximum threads available to the IAS Service. The default number of threads is one more than the number of CPUs of the machine running the IAS Service. If you are running multiple crawls concurrently, Oracle does not recommend increasing the value of this setting because it is possible for the system to over thread and consequently slow IAS performance. You can minimize the risk of over threading by reducing the value of this setting to no more than the number of CPUs plus 1 for each crawl running on a single IAS system.
`textExtractionConfig`	(Optional). Specify whether document conversion is enabled. If `textExtractionConfig` has a value of `true`, then IAS performs document conversion and stores the converted text as a property on the Endeca record.
`manipulatorConfigs`	(Optional). Specify any number of manipulators within a crawl configuration. If one or more `manipulatorConfig` elements are present, IAS passes each record to each manipulator for processing according to its `manipulatorConfig` settings. Manipulators execute in the order in which they are nested within `manipulatorConfigs`.

5. Configure the following settings that are specific to a SharePoint Object Model crawl:

Property Name	Property Description
`moduleId`	(Required). Specify the name of the IAS module for a SharePoint Object Model crawl. This value must be set to `Microsoft SharePoint Object Model`.

Property Name	Property Description
sharepointConnectorUrl	(Required). The SharePoint server name and port, such as http://sharepoint:10000. The sharepointConnectorUrl can only be set to the repository site or the home SharePoint site collection. The sharepointConnectorUrl cannot be set to a Document Library. sharepointConnectorUrl names are case sensitive.
httpChunkingEnabled	(Optional). Enable this property to use chunked encoding for HTTP messages. Enter true or false. The default value is true.
domain	(Required for NTLM authentication, otherwise optional.) In order to authenticate using NTLM, the domain name must be specified to log on to the server. For non-NTLM authentication, this is a convenience property for prepending the value of this property to the username property. The domain will be appended with a backslash separating it from the username. Endeca recommends specifying the domain only in the username property and not adding this property, for clarity.
strictSSLChecking	(Optional). Specify whether all SSL certificates are accepted, including self-signed certificates. If set to true, only trusted SSL certificates are accepted. The default value is false.
socketTimeout	(Optional). Specify a timeout value (in milliseconds) for content retrieval. The default value is 15 seconds (15000 milliseconds).
expandArchives	(Optional.) Specify whether to create a record for each archived entry and populate the record's properties.
seeds	(Optional). Specify a relative path to documents within a SharePoint site. This property can be multi-valued in order to specify each seed separately. If unspecified, the default seed is the root of the repository. For example: <moduleProperty> <key>seeds</key> <value>/Wiki Page Library</value> </moduleProperty>

6. Configure the settings to write crawl output to a Record Store. Although you can configure a crawl to write output to a file, writing to a Record Store is recommended and also the default. (To write to a file, see *Writing crawl output to a file on page 63*.)

Option	Description
`moduleId`	(Required). Specify the output type for a crawl. Specify an `id` of `Record Store` if the crawl is writing to a Record Store or specify `File System` if the crawl is writing to a file.
`isPortSsl`	(Required). Specify whether to use SSL when connecting to the Record Store instance. A value of `true` uses HTTPS and treats the `port` property as an SSL port. A value of `false` uses HTTP and treats `port` as a non-SSL port. Specify `false` if you enabled redirects from a non-SSL port to an SSL port.
`host`	(Required). Specify the fully qualified name of the host running the Record Store instance. The default value is `localhost`.
`port`	(Required). Specifies the port of the Endeca IAS Service running the Record Store instance. The default value is `8510`.
`instanceName`	(Optional). Specify the Record Store instance name to write to. If unspecified, this defaults to the same value as the `crawlId`.
`isManaged`	(Optional). Specify whether the Record Store is managed. If you disable the `isManaged` property by setting it to `false`, a Record Store instance is not created when you create the crawl. The default value is `true`.

7. Save and close the crawl configuration file.
8. Run the `createCrawls` task of `ias-cmd` to upload the crawl configuration file to IAS. See *Creating crawls on page 92*.

SharePoint versions supported by a SharePoint Object Model crawl

A SharePoint Object Model crawl supports the following versions of SharePoint:

- Microsoft Office SharePoint Server 2007 (MOSS 2007)
- Windows SharePoint Services 3.0 (WSS 3.0)
- SharePoint Server 2010
- SharePoint Foundation Server 2010

The crawl supports Basic authentication and NTLM authentication (Integrated Windows Authentication). All NTLM variations are supported, including LM, NTLM, LMv2, and NTLMv2.

Installing a SharePoint solution on the SharePoint server

Before you can configure the SharePoint Object Model crawl, you must install a SharePoint solution on the SharePoint server. This task is not required if you are using the SharePoint Web Services crawl.

The IAS installation redistributes a SharePoint solution. The SharePoint solution contains an agent (a SOAP Web service assembly) that allows the crawl to communicate with the SharePoint server.

The user installing the SharePoint solution must have the following roles in SharePoint:

- Farm Administrator
- Site Collection Administrator
- db_owner (content database of the administration site)

The SharePoint solution installs the following assemblies into the global assembly cache:

- `Microsoft.Web.Services2.dll`
- `Entropysoft.Sharepoint.WebService.dll`
- `Entropysoft.WebConfModif.dll`

To install the SharePoint solution:

1. Navigate to `<install path>\IAS\<version>\cms\sharepoint-om`.
2. Copy `EntropySoft-SharePoint-Conn-Setup.exe` to a temporary directory accessible from the SharePoint server.
3. Start the Windows SharePoint Services Administration service on the SharePoint server.
4. Double-click `EntropySoft-SharePoint-Conn-Setup.exe`. The SharePoint installation program starts.
5. Click **Yes**.
6. Click **Next** to continue.

   The installation program performs a number of preliminary checks. If one of these checks fail, correct the problem and restart the installation program to continue.
7. Accept the EntropySoft license agreement and click **Next** to continue.
8. On the Deployment Targets screen, select the SharePoint Web site(s) to deploy the crawl to. You may need to select multiple Web sites if you have a SharePoint server farm.

   There is normally no need to deploy the crawl to SharePoint administrative sites.
9. Click **Next** to continue.
10. Click **Close**.

The installation program copies the files and deploys the crawl to all members of the selected SharePoint farm.

After installation, you can verify that the SharePoint solution has been correctly deployed to the server, or server farm, by connecting to SharePoint Central Administration, clicking the Operations tab, then selecting Solution management in the Global Configuration section.

Additional configuration notes for a SharePoint Object Model crawl

Note the following when configuring a SharePoint Object Model crawl.

- When crawling lists, the SharePoint crawl ignores SharePoint filters and views and crawls all items in the list.

Configuring sharepointdata and seeds

- Specify each site collection as its own `sharepointConnectorUrl` configuration property rather than as a seed. IAS will only crawl a site collection if it is specified in the `sharepointConnectorUrl`. For example, a `sharepointConnectorUrl` of `http://sharepoint:1000/engineering` crawls the site collection named `engineering`. In some cases, a site collection contains a prefix name. If a prefix name exists, it must be specified in the `sharepointConnectorUrl`. This configuration requirement means that if you want to crawl multiple site collections, you should create a SharePoint crawl for each site collection. This is necessary because each site collection has its own independent scope for document types, groups, security settings, user accounts, and so on.
- When crawling MySites, specify the `sharepointConnectorUrl` as `http://`*`host`*`:`*`port`*`/personal/`*`username`* rather than `http://`*`host`*`:`*`port`*`/MySite`. SharePoint treats MySites as separate repositories or site collections.
- The value `/personal/`*`username`* must be included in the `sharepointConnectorUrl` and not the seed. For example, when using the `sharepointConnectorUrl` `http://sharepoint:1000`, an invalid seed is `/personal/`*`username`*`/Shared Documents`.
- The `sharepointConnectorUrl` can be used to specify specific subsites and supports nesting. For example, `http://sharepoint:1000/subsite1/subsite2` is a valid `sharepointConnectorUrl`.
- Specify seeds as relative to the `sharepointConnectorUrl` site collection or repository. For example, when using the `sharepointConnectorUrl` `http://sharepoint:1000`, a valid seed is `/Shared Documents/Word Docs`.
- Seed URLs are automatically encoded; do not encode seed URLs. For example, do not use "%20" to denote spaces; use spaces if needed in the URL. When crawling lists, do not specify the seed as the navigation url (such as /Lists/ListName); simply specify /ListName.
- IAS does not return content when crawling a `default.aspx` file. The content in `default.aspx` is made up of Web parts to other objects. The file is effectively a view on content but is not part of a site's or collection's actual content.

Required permissions in SharePoint 2007 and SharePoint 2010

- The minimum requirement for a SharePoint 2007 user account is the Read permission level. The Read permission level includes the following permissions: View Items, Open Items, View Versions, Create Alerts, View Application Pages, Use Self-Service Site Creation, View Pages, Browse User Information, Use Remote Interfaces, Use Client Integration Features, and Open. If there is additional content you want to crawl that is not accessible with that permission level, the user account or the content may need additional permissions.
- When crawling a SharePoint repository, users need the Enumerate Permissions setting. This permission can be added as an advanced permission setting in SharePoint, or you can set it as part of the default Full Control permission level.

- If a user has the Full Control or Manage Lists permission level, the crawl creates records for all content items including Galleries. (A record based on a Gallery item typically does not contain content that should be available in a search application.)

Limitations related to permissions

The IAS Server does not crawl certain SharePoint constructs due to permission limitations imposed by SharePoint Web services:

- The IAS Server logs an `InvalidCredentialsException` when crawling Topics or News constructs. No records are output for these constructs and the crawler will continue its processing as normal.

Limitations crawling galleries

A SharePoint Object Model crawl does not support crawling the following galleries:

- Web Part Gallery
- Site Template Gallery
- List Template Gallery
- Master Page Gallery

Limitations when using SharePoint 2007 and SharePoint Services 3.0

These limitations apply:

- Target audiences are not supported. There is no way to return the target audience of an item.
- Audience filtering is not supported.
- Content types are not supported.
- The `NoCrawl` property for lists and sites is not available.

Permission mapping for SharePoint Object Model properties

The following table shows the mapping between SharePoint permissions and the resulting Endeca record properties that are produced.

SharePoint Site permission	Endeca record properties
ViewPages	`Endeca.CMS.AllowReadProperties`

SharePoint List permission	Endeca record properties
ViewListItems	`Endeca.CMS.AllowReadProperties`

SharePoint permission	Endeca record properties
ViewListItems	`Endeca.CMS.AllowReadProperties` `Endeca.CMS.AllowReadContent`

Uninstalling the SharePoint solution from the SharePoint server

To uninstall the SharePoint solution from the SharePoint server:

1. Start the Windows SharePoint Services Administration service on the SharePoint server.
2. Go to the SharePoint 3.0 Central Administration console.
3. Click the **Operations** tab.
4. Under Global Configuration, click **Solution Management**.
   You should see `entropysoft.sharepoint.webservice.wsp` in the list of installed solutions.
5. Click `entropysoft.sharepoint.webservice.wsp`.
6. Click **Retract Solution**.
7. Ensure that **Now** is selected under When to retract solution.
8. Click **OK** to retract the solution now.
9. Refresh the Solution Management page until the status of the `entropysoft.sharepoint.webservice.wsp` solution is Not Deployed.
10. Click `entropysoft.sharepoint.webservice.wsp`.
11. Click **Remove Solution** and click **OK** in the confirmation prompt.

Creating a SharePoint Web Services crawl

You create a SharePoint Web Services crawl by making a copy of the default crawl configuration file for a SharePoint Web Services crawl and modifying the settings in the file as appropriate for your environment.

To create a SharePoint Web Services crawl:

1. In your IAS installation, locate the default crawl configuration files provided in `<install path>\IAS\<version>\sample\crawlConfigFiles`.
2. Make a copy of `sharePointWSCrawl.xml`, rename the file for your environment, and save it to a local directory.
3. Open the new crawl configuration file in a text editor.
4. Configure the settings that are common to all crawl types:

Option	Description
`crawlId`	(Required). Specify a unique name to distinguish the crawl from others in IAS. The `crawlId` can contain alphanumeric characters, underscores, dashes, and periods. All other characters are invalid for a `crawlId`.

Option	Description
`unavailableIncrementalSwitchesToFullCrawl`	(Required). Specify a Boolean value to indicate whether IAS should switch from running an incremental crawl to running a full crawl in cases where it is not possible to run an incremental crawl. A value of `true` instructs IAS to run a full crawl if it is not possible to run an incremental crawl. A value of `false` instructs IAS to abort the incremental crawl and throw a `FullCrawlRequiredException` indicating why the incremental crawl could not run. The default value depends on the `outputConfig` type. If the `outputConfig` is set to `Record Store`, then the default value is `true`. If the `outputConfig` is set to `File System`, then the default value is `false`.
`crawlThreads`	(Required). Specify the maximum threads available to the IAS Service. The default number of threads is one more than the number of CPUs of the machine running the IAS Service. If you are running multiple crawls concurrently, Oracle does not recommend increasing the value of this setting because it is possible for the system to over thread and consequently slow IAS performance. You can minimize the risk of over threading by reducing the value of this setting to no more than the number of CPUs plus 1 for each crawl running on a single IAS system.
`textExtractionConfig`	(Optional). Specify whether document conversion is enabled. If `textExtractionConfig` has a value of `true`, then IAS performs document conversion and stores the converted text as a property on the Endeca record.
`manipulatorConfigs`	(Optional). Specify any number of manipulators within a crawl configuration. If one or more `manipulatorConfig` elements are present, IAS passes each record to each manipulator for processing according to its `manipulatorConfig` settings. Manipulators execute in the order in which they are nested within `manipulatorConfigs`.

5. Configure the following settings that are specific to a SharePoint Web Services crawl:

Option	Description
`moduleId`	(Required). Specify the name of the IAS module for a SharePoint Web Services crawl. This value must be set to `Microsoft SharePoint Web Services`.
`siteUrl`	(Required). Specify the SharePoint server name and port, such as `http://sharepoint:10000`. The `siteUrl` can only be set to the repository site or the home SharePoint site collection. The `siteUrl` cannot be set to a Document Library. `siteUrl` names are case sensitive.
`handleGenericLists`	(Optional). Specify whether to support additional Sharepoint lists such as Issues, Wiki, Surveys, custom lists. By default, the crawl manages document libraries. The default value is `true`.

Option	Description
`httpChunkingEnabled`	(Optional). Specify whether to use chunked encoding for HTTP messages. The default value is `true`.
`domain`	(Required for NTLM authentication, otherwise optional.) Specify the domain name to log on to the server using NTLM. For non-NTLM authentication, this is a convenience property for prepending the value of this property to the username property. The domain will be appended with a backslash separating it from the username. Endeca recommends specifying the domain only in the username property and not adding this property, for clarity.
`strictSSLChecking`	(Optional). Specify whether all SSL certificates are accepted, including self-signed certificates. If set to `true`, only trusted SSL certificates are accepted. The default value is `false`.
`expandArchives`	(Optional.) Specify whether to create a record for each archived entry and populate the record's properties.

`seeds`	(Optional). Specify a relative path to documents within a SharePoint site. This property can be multi-valued in order to specify each seed separately. If unspecified, the default seed is the root folder of the repository. For example:
	`<moduleProperty>`
	`<key>seeds</key>`
	`<value>/Wiki Page Library</value>`
	`</moduleProperty>`

6. Configure the settings to write crawl output to a Record Store. Although you can configure a crawl to write output to a file, writing to a Record Store is recommended and also the default. (To write to a file, see *Writing crawl output to a file on page 63*.)

Option	Description
`moduleId`	(Required). Specify the output type for a crawl. Specify an `id` of `Record Store` if the crawl is writing to a Record Store or specify `File System` if the crawl is writing to a file.
`isPortSsl`	(Required). Specify whether to use SSL when connecting to the Record Store instance. A value of `true` uses HTTPS and treats the `port` property as an SSL port. A value of `false` uses HTTP and treats `port` as a non-SSL port. Specify `false` if you enabled redirects from a non-SSL port to an SSL port.
`host`	(Required). Specify the fully qualified name of the host running the Record Store instance. The default value is `localhost`.
`port`	(Required). Specify the port of the Endeca IAS Service running the Record Store instance. The default value is `8510`.

Option	Description
`instanceName`	(Optional). Specify the Record Store instance name to write to. If unspecified, this defaults to the same value as the `crawlId`.
`isManaged`	(Optional). Specify whether the Record Store is managed. If you disable the `isManaged` property by setting it to `false`, a Record Store instance is not created when you create the crawl. The default value is `true`.

7. Save and close the crawl configuration file.
8. Run the `createCrawls` task of `ias-cmd` to upload the crawl configuration file to IAS. See *Creating crawls on page 92*.

SharePoint versions supported by a SharePoint Web Services crawl

A SharePoint Web Services crawl supports the following versions of SharePoint:

- SharePoint Portal Server 2003 (SPS 2003)
- Windows SharePoint Services 2.0 (WSS 2.0)
- Windows SharePoint Services 3.0 (WSS 3.0)
- Microsoft Office SharePoint Server 2007 (MOSS 2007)
- SharePoint Server 2010
- SharePoint Foundation Server 2010

A crawl supports Basic authentication and NTLM authentication (Integrated Windows Authentication). All NTLM variations are supported, including LM, NTLM, LMv2, and NTLMv2.

Additional configuration notes for a SharePoint Web Services crawl

Note the following when configuring a SharePoint Web Services crawl.

- A SharePoint crawl uses the standard SharePoint Web services API. To retrieve the content of a document, the crawl directly uses HTTP or HTTPS GET.
- When crawling lists, a crawl ignores filters and views and crawls all items in the list.
- Due to a SharePoint 2003 Web services defect, crawling Meeting Workspaces causes the server to queue child pages such as Workspace Pages that do not exist, which in turn causes an Exception error message during a crawl. The crawl finishes processing documents correctly in the Document Library.

Configuring siteUrl and seeds

- When crawling MySites, specify the `siteUrl` as `http://host:port/personal/username` rather than `http://host:port/MySite`. SharePoint treats MySites as separate repositories or site collections.
- The value `/personal/username` must be included in the `siteUrl` and not the seed. For example, when using the `siteUrl` `http://sharepoint:1000`, an invalid seed is `/personal/username/Shared Documents`.

- The `siteUrl` can be used to specify specific subsites and supports nesting. For example, `http://sharepoint:1000/subsite1/subsite2` is a valid `siteUrl`.
- Specify seeds as relative to the `siteUrl` site collection or repository. For example, when using the siteUrl `http://sharepoint:1000`, a valid seed is `/Shared Documents/Word Docs`.
- Seed URLs are automatically encoded; do not encode seed URLs. For example, do not use "%20" to denote spaces; use spaces if needed in the URL. When crawling lists, do not specify the seed as the navigation url (such as /Lists/ListName); simply specify /ListName.
- IAS does not return content when crawling a `default.aspx` file. The content in `default.aspx` is made up of Web parts to other objects. The file is effectively a view on content but is not part of a site's or collection's actual content.

Required permissions in SharePoint 2003

- The minimum requirement for a SharePoint 2003 user account is membership in the Reader Site Group. The Reader Site Group includes the following permissions: View Area, View Pages, and Search. If there is additional content you want to crawl that is not accessible with that permission, the user account or the content may need additional permissions.

Required permissions in SharePoint 2007

- The minimum requirement for a SharePoint 2007 user account is the Read permission level. The Read permission level includes the following permissions: View Items, Open Items, View Versions, Create Alerts, View Application Pages, Use Self-Service Site Creation, View Pages, Browse User Information, Use Remote Interfaces, Use Client Integration Features, and Open. If there is additional content you want to crawl that is not accessible with that permission level, the user account or the content may need additional permissions.
- When crawling a SharePoint repository, users need the Enumerate Permissions setting. This permission can be added as an advanced permission setting in SharePoint, or you can set it as part of the default Full Control permission level.
- If a user has the Full Control or Manage Lists permission level, the crawl creates records for all content items including Galleries. (A record based on a Gallery item typically does not contain content that should be available in a search application.)

Limitations related to permissions

The IAS Server does not crawl certain SharePoint constructs due to permission limitations imposed by SharePoint Web services:

- The IAS Server logs an `InvalidCredentialsException` when crawling Topics or News constructs. No records are output for these constructs and the crawler will continue its processing as normal.
- This applies to SharePoint 2003 only. The IAS Server does not directly crawl a SharePoint Area when specified as a `siteUrl`. An Area must be specified as a seed and not as a `siteUrl`. However, the IAS Server can crawl specific content within an Area, such as a Document Library, if that content is specified as a seed. The IAS Server cannot retrieve permissions for a SharePoint Area.

Limitations when using SharePoint 2007, SharePoint Services 3.0, and SharePoint 2010

These limitations apply:

- SharePoint 2007 supports security at the item level but the crawl's Web services do not. An item's returned permissions are the permissions of the list containing the item.
- Target audiences are not supported. There is no way to return the target audience of an item.
- Audience filtering is not supported.
- Content types are not supported.
- The `NoCrawl` property for lists and sites is not available.

Permission mapping for SharePoint Web Services properties

The following table shows the mapping between SharePoint permissions and the resulting Endeca record properties that are produced.

SharePoint permission	Endeca record properties
View Items	`Endeca.CMS.AllowReadProperties` `Endeca.CMS.AllowReadContent`

About filters

Filters define which folders and files are included and excluded when IAS runs a crawl. You specify filters using `ias-cmd` or in the IAS Server API.

Filters perform matching operations against a property on an Endeca record and either include or exclude the record based on the filter's evaluation. You specify both the Endeca property to evaluate and the data type and expression to match against that property.

If an include filter matches (evaluates to true) against a property, then that record is included in the record set. If an exclude filter matches (evaluates to true) against a property, then that record is excluded from the record set.

Filters perform matching based on the following data types:

- Date - a date value against which files and folders can be filtered.
- Long - a long value to compare against a numerical property.
- String - a string value to compare against a string property. String filters are either regex or wildcard.

Regex - a regular expression value to compare against a string property. The matching evaluation is one of equality: the string either matches the expression or it does not match.

Wildcard - a wildcard expression value to compare against a string property. The wildcard matcher uses the question-mark (?) character to represent a single wildcard character and the asterisk (*) to represent multiple wildcard characters. The matching evaluation is one of equality: the string either matches the expression or it does not match. Also, there must be either all include filters or all exclude filters per property.

The following sub-headings define the way filters operate:

One filter per Endeca property per file or folder (unless a wildcard)

You can create one filter per Endeca property that applies to a file and one filter per Endeca property that applies to a folder, unless you are creating a wildcard filter. (You can create any number of wildcard filters for an Endeca property.) This also means you can create a file filter and a folder filter that apply to the same Endeca property.

AND'ing and OR'ing

- If you create multiple filters on a single property (wildcard filters), they are logically OR'ed during filter evaluation.
- Filters across properties are logically AND'ed during filter evaluation. Remember that AND means that all filters must match in order for a record to be included or excluded.

Include and exclude filters

- Include filters may apply to either folders or files.
- Exclude filters apply to only to folders.

Filter precedence

If you use both include and exclude filters, exclude filters take precedence. For example, if a `test.doc` file was recently modified and you add an include filter for `test.doc` but then add an exclude filter that excludes all recently modified files, the `test.doc` will not be crawled.

Missing properties on a record

Filters require an Endeca property to match against. In cases where the property for a filter does not exist on a record, the behavior varies depending on whether the filter is an include or an exclude.

- If the filter is an include and the property does not exist on a record, the record is excluded.
- If the filter is an exclude and the property does not exist on a record, the record is included.

Unfilterable properties

Do not use the `Endeca.Document` properties for filter matching. These properties are generated by the IAS Document Conversion Module *after* a file or folder is crawled and filtered, and therefore cannot be used to filter files or folders.

Case sensitivity

Regex filters are case sensitive by default (however, you may construct a regular expression that is case insensitive). Wildcard filters are case insensitive.

Setting document conversion options

You can change the behavior of the IAS Document Conversion Module for identifying fallback format, file identification, and extracting hidden text. You change the default document conversion behavior by specifying options via JVM property names and values.

The options are:

- `stellent.fallbackFormat` determines the fallback format, that is, what extraction format will be used if the IAS Document Conversion Module cannot identify the format of a file. The two valid settings are `ascii8` (unrecognized file types are treated as plain-text files, even if they are not plain-text) and `none` (unrecognized file types are considered to be unsupported types and therefore are not converted). Use the `none` setting if you are more concerned with preventing many binary and unrecognized files from being incorrectly identified as text. If there are documents that are not being properly extracted (especially text files containing multi-byte character encodings), it may be useful to try the `ascii8` option.
- `stellent.fileId` determines the file identification behavior. The two valid settings are `normal` (standard file identification behavior occurs) and `extended` (an extended test is run on all files that are not identified). The `extended` setting may result in slower crawls than with the `normal` setting, but it improves the accuracy of file identification.
- `stellent.extractHiddenText` indicates whether to convert hidden text stored in a content item. Hidden text may include text produced by optical character recognition (OCR) software in addition to other types of hidden text. Specifying `true` for `stellent.extractHiddenText` converts any hidden text stored in the content item. Specifying `false` does not convert hidden text.

Default values for the options

The default settings for the options are listed in the following table.

Option	Defaults
`stellent.fallbackFormat`	none
`stellent.fileId`	extended
`stellent.extractHiddenText`	If unspecified, the default value is `false`.

Setting the options

You set the text extraction options as parameters to the Java Virtual Machine (JVM), via the Java `-D` option. To set the fallback format, use one of these two parameters:

```
-Dstellent.fallbackFormat=ascii8
-Dstellent.fallbackFormat=none
```

To set the file identication behavior, use one of these two parameters:

```
-Dstellent.fileId=normal
-Dstellent.fileId=extended
```

To enable the extraction of hidden text, use this parameter:

```
-Dstellent.extractHiddenText=true
```

To pass these parameters to the JVM, use the `-JVM` flag when you run the Endeca IAS Service script or add JVM arguments to the script itself. Note that for Windows machines, the parameters should be quoted if they contain equal signs, as in this example:

```
ias-service -JVM "-Dstellent.fallbackFormat=ascii8"
```

Note that when using the `-JVM` flag, it must be the last flag on the command line.

Configuring document conversion filters

You can configure a set of filters to apply only to document conversion. These filters either convert or do not convert files of a specified size or type by using include or exclude filters. The include and exclude filters apply only to document conversion.

The document conversion filters perform matching against any Endeca property and include or exclude a file from the document conversion process but still produce an Endeca record for the file. If a file is included for document conversion, the corresponding Endeca record has an `Endeca.Document.Text` property.

Data source extensions built using the IAS Extension API do not support document conversion filters. Any changes you make to `DocumentConversionFilters.xml` are not applied to data source extensions. Also, IAS does not apply document conversion filters to archive files. However, you can enable the **Expand archives** option and then IAS can process the extracted content.

You configure document conversion filters by modifying *<install path>*`\IAS\workspace\conf\DocumentConversionFilters.xml`. These document conversion filters apply to all data sources that have document conversion enabled; the filters do not apply on a per-data source basis.

This file has sections for CMS data sources and file system crawls. Within a CMS or file section, the file has a section for include filters and for exclude filters. Here is an example snippet of the structure of the file:

```
<cmsCrawlDocumentConversionFilters>
        <includeFilters>
    ...
        </includeFilters>
        <excludeFilters>
    ...
        </excludeFilters>
    </cmsCrawlDocumentConversionFilters>

<fileCrawlDocumentConversionFilters>
        <includeFilters>
    ...
        </includeFilters>
        <excludeFilters>
    ...
        </excludeFilters>
</fileCrawlDocumentConversionFilters>
```

Inside the `includeFilters` and `excludeFilters` sections are the filters themselves. Each is indicated by a `filter` element. For example, this snippet shows a regular expression filter for file system crawls that includes all files of the types listed:

```
<fileCrawlDocumentConversionFilters>
        <includeFilters>
            <filter xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="regexFilter">
                <scope>FILE</scope>
                <propertyName>Endeca.FileSystem.Name</propertyName>
                <regex>^(?i:.*\.(?:txt|html?|rtf|docx?|xlsx?|
      pptx?|pdf|sxi|sxc|sxw|shw|qpw|wpd|xml))$</regex>
            </filter>
```

```
...
        </includeFilters>
```

Each filter is made up of the following XML:

Element or attribute	Description
`type` attribute of `filter`	The value of `type` can be either `regexFilter`, `longFilter`, `dateFilter`, or `wildcardFilter`.
`scope`	The value of scope should always be set to `FILE`. Document conversion filters apply only to files; do not modify this value.
`propertyName`	The value of `propertyName` specifies the Endeca property on which you want the filter to perform matching operations. Common filter properties are `Endeca.FileSystem.Name`, and `Endeca.FileSystem.Extension`.
`operator`	The `operator` for `type="longFilter"` performs numeric comparisons using any of the following values: `EQUAL`, `GREATER`, `GREATER_EQUAL`, `LESS`, `LESS_EQUAL`, and `NOT_EQUAL`. The `operator` for `type="dateFilter"` performs comparisons against date time values using either `BEFORE` or `AFTER`.
`regexFilter`	Specifies a regular expression to compare against the specified property.
`longFilter`	Specifies a numeric value to compare against the specified property.
`dateFilter`	Specifies a date against which files can be filtered.
`wildcardFilter`	Specifies a wildcard to match against a specified property. The wildcard matcher uses the question-mark (?) character to represent a single wildcard character and the asterisk (*) to represent multiple wildcard characters. Matching is case insensitive: this is not configurable (If case sensitivity is required, consider using a regular expression).

Like other types of filters, document conversion filters cannot have multiple filters with the same `propertyName` unless the filters are `wildcardFilter`.

Note: Mime type properties depend on the data source and you may need to check that you add the correct mime type to your filters. Also, some CMS data sources may not produce an `Endeca.CMS.ContentLength` property and therefore, you may not be able to filter those files by size.

To configure document conversion filters:

1. Navigate to `IAS\workspace\conf\`, and open `DocumentConversionFilters.xml` in a text editor.
2. Add include and and exclude filters according to the syntax described above.
   The `DocumentConversionFilters.xml` file contains default filters that you can use as examples. These filters are include filters for the most common document types such as txt, html, rtf, doc, pdf, and so on.

3. Comment out any filters that you do not want applied.
4. Save and close the file.
5. Restart the Endeca IAS Service.
6. Optionally, perform a full crawl of the data source after configuring filters.

The file is validated against the schema and if there are validation errors, the Endeca IAS Service does not start. (This is logged in the IAS Service log file.) The file must conform to `DocumentConversionFilters.xsd`. If the file is missing, then IAS Server converts all documents by default.

Adding a Filtering Script manipulator to a crawl

A Filtering Script manipulator runs an inline BeanShell script that filters Endeca records from crawl output. You add and configure a Filtering Script manipulator inside a `manipulatorConfig` element of the crawl configuration file, and you identify the manipulator with a `moduleId` of `com.endeca.ias.manipulator.FilterScriptManipulator`.

The manipulator has access to the methods in the `Record` and `PropertyValue` classes (i.e. `com.endeca.eidi.record` and `com.endeca.eidi.record.PropertyValue`). For details about the methods in these classes, see the *IAS Record Store API Reference (Javadoc)* installed in `IAS\<version>\doc\recordstore-javadoc`. The manipulator also has access to the methods in the `Logger` class in `org.slf4j.Logger`. Other classes may be imported as necessary.

The manipulator supports BeanShell 2.0b4 and later. For more information about BeanShell scripting, see http://www.beanshell.org.

To add a Filtering Script manipulator to a crawl:

1. Open the crawl configuration file in a text editor.
2. Locate the `manipulatorConfigs` element.
   (This element is a container for one or more `manipulatorConfig` components.)
3. Inside the `manipulatorConfigs` element, add the following XML to specify a default Filtering Script manipulator:

```
<manipulatorConfig>
                <moduleId>
                    <id>com.endeca.ias.manipulator.FilterScriptManipulator</id>
                </moduleId>
                <moduleProperties>
                    <moduleProperty>
                        <key>scriptSource</key>
                        <value>logger.info("Processing Record:" + record.toString())
;filePropertyValue = new PropertyValue("Endeca.FileSystem.IsDirectory", "true")
;return !record.hasPropertyValue(filePropertyValue);</value>
                    </moduleProperty>
                </moduleProperties>
                <id>defaultFilterScriptManipulator</id>
</manipulatorConfig>
```

4. Configure the following settings:

Option	Description
moduleId	(Required). Specify the ID for a Filtering Script manipulator. This value must be set to com.endeca.ias.manipulator.FilterScriptManipulator.
scriptSource	(Required). Specify the inline BeanShell script to perform filtering operations. The example script, provided in the scriptSource element, evaluates a record and then returns a Boolean to indicate whether a particular record is included or excluded from the acquisition output. You can modify the inline script as appropriate for your acquisition. A return value of true includes a record and false excludes the record from the crawl output. If the script does not return a Boolean, the crawl fails.
id	(Required). Specify the ID for this instance of a Filtering Script manipulator.

5. Save and close the crawl configuration file.
6. Run either the createCrawls task or the updateCrawls of ias-cmd to upload the crawl configuration file to IAS.

Adding a Modifying Script manipulator to a crawl

A Modifying Script manipulator runs an inline BeanShell script that modifies Endeca records. You add and configure a Modifying Script manipulator inside a manipulatorConfig element of the crawl configuration file, and you identify the manipulator with a moduleId of com.endeca.ias.manipulator.ModifierScriptManipulator.

The manipulator has access to the methods in the Record and PropertyValue classes (i.e. com.endeca.eidi.record and com.endeca.eidi.record.PropertyValue). For details about the methods in these classes, see the *IAS Record Store API Reference (Javadoc)* installed in IAS\<version>\doc\recordstore-javadoc. The manipulator also has access to the methods in the Logger class in org.slf4j.Logger. Other classes may be imported as necessary.

The manipulator supports BeanShell 2.0b4 and later. For more information about BeanShell scripting, see http://www.beanshell.org.

To add a Modifying Script manipulator to a crawl:

1. Open the crawl configuration file in a text editor.
2. Locate the manipulatorConfigs element.
   (This element is a container for one or more manipulatorConfig components.)
3. Inside the manipulatorConfigs element, add the following XML to specify a default Modifying Script manipulator:

```
<manipulatorConfig>
                <moduleId>
                    <id>com.endeca.ias.manipulator.ModifierScriptManipulator</id>
                </moduleId>
                <moduleProperties>
                    <moduleProperty>
                        <key>scriptSource</key>
```

```
                        <value>idPropertyValue = record.getPropertySingleValue("Endeca.Id");
                        record.addPropertyValue(new PropertyValue("New.Prop", "IAS"));
                        logger.info("Processed Record:" + idPropertyValue.Value);</value>
                    </moduleProperty>
                </moduleProperties>
                <id>addIASPropertyManipulator</id>
</manipulatorConfig>
```

4. Configure the following settings:

Option	Description
`moduleId`	(Required). Specify the ID for a Modifying Script manipulator. This value must be set to `com.endeca.ias.manipulator.ModifierScriptManipulator`.
`scriptSource`	(Required). Specify the inline BeanShell script to perform modifying operations. The example script, provided in the `defaultValue` element, adds a property to each record being crawled and then logs that addition. You can modify the inline script as appropriate for your crawl.
`id`	(Required). Specify the ID for this instance of a Modifying Script manipulator.

5. Save and close the crawl configuration file.
6. Run either the `createCrawls` task or the `updateCrawls` of `ias-cmd` to upload the crawl configuration file to IAS.

Modifying a crawl

Modifying a crawl may be useful if you want to re-configure any settings. For example, if you modify a crawl that writes to an output file, you could re-configure a different output directory, different file format, and so on. If you modify a crawl that writes to a Record Store instance, you could re-configure a different host machine, or use SSL, and so on.

To modify a crawl:

1. Start a command prompt, navigate to `<install path>\IAS\<version>\bin`.
2. Run the `getCrawl` task of the `ias-cmd` and specify the `-f` option with an argument that specifies a path for the crawl configuration file and also specify the `-id` option with the ID of the crawl. Optionally, you may want to specify the `-d` option to write default values for the configuration properties.
   For example, this command identifies a crawl named `itldocset`, gets its configuration, and writes it to `C:\tmp\crawlConfig.xml`.

   ```
   ias-cmd getCrawl -d -f C:\tmp\crawlConfig.xml -id itldocset
   ```

3. Open the crawl configuration file in a text editor and modify as necessary. The configuration settings vary depending on the crawl types described previously in this chapter.
4. Save and close the crawl configuration file.
5. Run the `updateCrawls` task of the `ias-cmd` and specify the `-f` option with an argument that specifies the name for the crawl configuration file you modified in the previous steps.
   For example, this command creates a crawl named `itldocset`.

   ```
   ias-cmd updateCrawls -f C:\tmp\crawlConfig.xml
   ```

If the task succeeds, the console displays a message similar to the following:

```
Updated crawl itldocset
```

Writing crawl output to a file

In some cases, you may want to reconfigure a crawl to write output to a file rather than to a Record Store instance. For example, this may be useful if you want to examine the output as XML before further processing, or you may want to store the output in a version control system rather than in a Record Store.

The procedure requires a crawl configuration file with `<outputConfig>` settings that specify `File System` as the `<moduleId>`. The other sub-elements of `<outputConfig>` are additional configuration about the output file itself such as whether compression is enabled, a file prefix name, the path to the output file, and so on.

To write crawl output to a file:

1. Get the crawl configuration file:
   (a) Open a command prompt window and navigate to `<install path>\IAS\<version>\bin`.
   (b) Run the `getCrawl` task of `ias-cmd` and specify the `-id` and the `-f` flags. For details, see *Getting a crawl on page 95*.
2. Open the following crawl configuration file in a text editor.
3. Configure the output settings to write to a file:

Option	Description
`moduleId`	(Required). Specifies an `id` of `File System`.
`outputXml`	(Optional). Specifies whether to write the records as XML or binary. A value of `true` writes a single XML output file of records. A value of `false` writes binary files of records. The default value is `false`.
`outputCompressed`	(Optional). Specifies whether to compress the output file or not. Specifying `true` compresses the output. The default is `true`.
`outputPrefix`	(Optional). Specifies an output prefix to the file name. The default prefix is `CrawlerOutput`.

Option	Description
`outputDirectory`	(Optional). Specifies an output directory for the output file using `outputDirectory`. The default value of `outputDirectory` is `output`. The default name of *`crawlID`* is used to create a subdirectory for each crawl. This ensures each crawl has a unique subdirectory for its output. For example, if you use the default value for `outputDirectory` and have a *`crawlID`* of `FileSystemCrawl`, the resulting directory structure is `\`*`IASServerWorkspace`*`\output\FileSystemCrawl\`. Example of syntax on Windows: `C:\Oracle\Endeca\IAS\workspace\output\FileSystemCrawl`

4. Save and close the crawl configuration file.
5. Run the `updateCrawls` task of `ias-cmd` to upload the crawl configuration file to IAS. For details, see *Updating crawls on page 100*.

Example configuration for writing output to a file

This sample shows a crawl configuration that writes output to an output file. The `sourceConfig` elements are removed for simplicity.

```
<?xml version="1.0" encoding="UTF-8"?>

<configurations xmlns="http://endeca.com/eidi/ias/2011-12">
    <crawlConfig>

    ...

      <outputConfig>
        <moduleId>
          <id>File System</id>
        </moduleId>
        <moduleProperties>
          <moduleProperty>
            <key>outputXml</key>
            <value>true</value>
          </moduleProperty>
          <moduleProperty>
            <key>outputCompressed</key>
            <value>false</value>
          </moduleProperty>
          <moduleProperty>
            <key>outputPrefix</key>
            <value>CrawlerOutput</value>
          </moduleProperty>
          <moduleProperty>
            <key>outputDirectory</key>
            <value>C:\Oracle\Endeca\IAS\workspace\output\FileSystemCrawl</value>
          </moduleProperty>
        </moduleProperties>
      </outputConfig>
    </crawlConfig>
</configurations>
```

Chapter 4

Configuring a Record Store Instance

This section provides detailed information the IAS Record Store service and explains how to configure a Record Store instance.

About record generations

A set of records that has been committed to a Record Store instance is a record generation.

For example, if you perform a full crawl, all the records produced from the crawl are written to the Record Store and a commit is done. After the commit, the Record Store has one generation of records. A subsequent crawl, either full or incremental, results in a second generation of records.

Each record that is read in contains a unique ID. IAS uses that unique ID as the value of the `idPropertyName` Record Store configuration property.

If a record already exists with that unique ID during later IAS crawls, then the later version replaces the earlier one. This ensures that when you run an incremental crawl, you always get the latest version of any given record.

A record generation is removed from a Record Store instance by the `clean` task after the generation becomes stale. A stale generation is a generation that has been in a Record Store instance for a period of time that exceeds the value of the `generationRetentionTime` Record Store configuration property. A stale generation is retained in several exception cases:

- A generation is currently in use. This occurs because either a `READ` transaction or a `READ_WRITE` transaction is running.
- If there is only one generation in a Record Store instance, it is not removed, even if it is stale.
- A generation is marked as a last-read generation for a client.

About transactions

A transaction is an access operation in the Record Store by another component. Transactions provide a means to keep one operation isolated from another operation and allow each to operate independently.

In other words, one transaction can read while another is writing. Each transaction is either a `READ_WRITE` transaction or a `READ` transaction:

- `READ` transactions support only Read operations. There may be any number of `READ` transactions running simultaneously. Examples of Read operations are Integrator reading a record generation for a baseline update or an administrator using the Record Store Command-line utility with the `-c` flag to get the count (number of records) in a Record Store instance.
- `READ_WRITE` transactions support both Read and Write operations. There may be only one `READ_WRITE` transaction running at any time. An example of a Write operation is a crawler running a full crawl and writing the output to a Record Store instance.

Each transaction is assigned a transaction ID. When a transaction begins, the Record Store service logs an INFO message with the transaction type and ID, as in this example of performing a `READ` transaction (with an ID of 2) for a baseline update:

```
Started transaction 2 of type READ
```

An example of a Write transaction message would be the following:

```
Started transaction 3 of type READ_WRITE
```

Each transaction has a status, which is one of the following:

- `ACTIVE` – The transaction is currently active. For example, the transaction is in the middle of a Write operation.
- `COMMITTED` – The transaction has successfully finished. An INFO message of “Committed transaction” is logged to indicate this status.
- `COMMIT_FAILED` – A commit transaction failed. The only operation allowed on the transaction is a rollback.
- `ROLLED_BACK` – The transaction has been successfully rolled back. No further operations are allowed on the transaction.

The rules for transactions are as follows:

- Once a transaction has been committed or rolled back, additional operations that try to access the transaction will fail.
- Once a Read operation has ended, additional operations that try to access the read cursor will fail.
- Only one operation per transaction can run at a time.
- If a transaction is rolled back, then it cancels operations in progress.

About the last read generation for a client

A Record Store instance can save the last read generation and also track that generation for any number of unique clients. You specify a unique client by creating a client ID, which can be any string, such as

rsreader1, and then set a value to indicate the last-read generation for that client ID. If you are using Integrator, you can use the default client ID of Integrator.

There are two ways to set the last read generation for a client:

- Automatically, in an Integrator graph, by using a Record Store Reader component to read records from the Record Store. In the component, set **IAS Client ID** to specify the client ID for the generation that is being read in.
- Manually, by using the set-last-read-generation task of the Record Store command-line utility.

Here is an example using Integrator to process the records:

1. You run a full IAS crawl and it writes the records to a Record Store as Generation 1.
2. You run an Integrator graph using Generation 1. You set the Record Store Reader component with a **IAS Read Type** of Full Extract and a **IAS Client ID** of Integrator.
3. You run a second IAS crawl, either full or incremental, and store the records as Generation 2. Because both crawls use the same idPropertyName and the same seeds, some of the records in both generations are identical and the others are delta records (new, modified, or deleted records).
4. You run an Integrator graph using the delta records between Generation 1 and Generation 2. For this graph, you set the Record Store Reader component with a **IAS Read Type** of Incremental and a **IAS Client ID** of Integrator.
5. The Integrator graph processes the delta records.

To find out which client states are currently saved in a Record Store instance, use the list-client-states task of the Record Store command-line utility.

About deleted records

Any client of the Record Store, including Integrator, the IAS Server, the Web Crawler, the IAS API, and so on, can modify and delete records that are written to the Record Store. Clients either update or insert a record, delete a record, or delete all records. This topic describes the Endeca.Action property that the Record Store examines to determine whether to update or insert or delete records.

Deleting all records for a full crawl

A record that has *only* the Endeca.Action property set to DELETE (i.e., has no other properties) functions as a logical “Delete All” record. When the Record Store encounters such a record, the Record Store removes all records from a Record Store instance. This is useful when running a full crawl and you want to remove a generation of records before writing a new generation.

For example:

```
<RECORDS>
  <RECORD>
    <PROP NAME="Endeca.Action">
      <PVAL>DELETE</PVAL>
    </PROP>
  </RECORD>
  ...
</RECORDS>
```

Deleting records for an incremental crawl

If a record has an `Endeca.Action` property set to `DELETE`, the record is removed from a Record Store instance. This property setting is useful in an incremental crawl where files may have been modified or deleted since the last crawl.

If an incremental crawl does not find a file that is listed in the crawl history, the IAS Server treats that file as deleted. For each deleted file, a record is created that contains the location of the deleted file and an `Endeca.Action` property with a value of `DELETE`.

For renamed files, the file with the old name is treated as a deleted file while the file with the new name is treated as a new (added) file.

This example shows the record for a `TestPlan.doc` file that was deleted:

```
<RECORDS>
  <RECORD>
    <PROP NAME="Endeca.Action">
      <PVAL>DELETE</PVAL>
    </PROP>
    <PROP NAME="Endeca.FileSystem.Path">
      <PVAL>c:\endeca_test_docs\TestPlan.doc</PVAL>
    </PROP>
    <PROP NAME="Endeca.SourceType">
      <PVAL>FILESYSTEM</PVAL>
    </PROP>
    <PROP NAME="Endeca.SourceId">
      <PVAL>FileSystemSource</PVAL>
    </PROP>
  </RECORD>
  ...
</RECORDS>
```

Reading records marked with the `DELETE` property value

Any client of the Record Store, for example a Record Store Reader in an Integrator graph, can read from a Record Store instance and process records that are marked with the `Endeca.Action` property set to `DELETE`.

Configuring a Record Store instance

Each uniquely named Record Store instance has its own configuration settings. You can run the `get-configuration` task of the Record Store Command-line Utility to save the configuration settings to a file, or you can create the file manually.

You then modify the configuration properties in the file and then run the `set-configuration` task to apply the configuration changes to a particular Record Store instance. Changes to the properties take effect immediately.

Note: If you change the `btreePageSize`, `changePropertyNames`, `idPropertyName`, `jdbmSettings`, or `recordCompressionEnabled` properties, the Record Store deletes all stored data.

To configure a Record Store instance:

1. Start a command prompt and navigate to `<`*`install path`*`>\IAS\<version>\bin`.

2. Type `recordstore-cmd` and specify the `get-configuration` task with the name of a Record Store instance and the XML file name where you want to save the configuration settings.

   This Windows example gets the configuration for a Record Store named `productdata`:

   ```
   recordstore-cmd.bat get-configuration -a productdata -f C:\tmp\config.xml -n
   ```

3. In a text editor, open the configuration file and modify the property values as described in *Configuration properties for a Record Store instance on page 69*.
4. Save and close the configuration file.
5. In the command prompt window, type `recordstore-cmd` and specify the `set-configuration` task with the name of a Record Store instance and the XML file name that contains the configuration settings.

   This example sets the configuration for a Record Store named `productdata`:

   ```
   recordstore-cmd.bat set-configuration -a productdata -f C:\tmp\config.xml
   ```

Configuration properties for a Record Store instance

The Record Store supports the configuration properties listed in the table below.

Configuration Property	Value
`btreePageSize`	Oracle Internal Use. Oracle does not recommend modifying this property. The number of children per btree node. This value must be greater than `0`. The default is 100. The Record Store validates that the `btreePageSize` property is greater than `0`. If this property is not correctly set, `RecordStore.setConfiguration()` throws a `RecordStoreConfigurationException`.

Configuration Property	Value
changePropertyNames	Specifies which record properties to examine when determining whether a record has changed. The changePropertyNames configuration property is useful because it allows you to specify exactly which record properties are evaluated to determine if that record has changed between crawls (in other words, if the record is different from the previous generation's record in the Record Store instance). If not specified, the value of changePropertyNames defaults to all the properties on a record. If you choose to specify the value changePropertyNames, here are several suggested properties: • File system crawls: Endeca.Document.Text, Endeca.FileSystem.ModificationDate • CMS crawls: Endeca.Document.Text, Endeca.CMS.ModificationDate • Web crawls: Endeca.Document.Text, Endeca.Web.Last-Modified If you are gathering native file system properties for file system crawls, you can also use the ACL properties as change properties.
cleanerInterval	Specifies how often (in hours) the Record Store Service checks for stale generations of records in a Record Store. (A stale generation is defined by the generationRetentionTime property.) This value of cleanerInterval must be greater than or equal to 0. If not specified, the value defaults to 1 hour. Fractional values (like 0.1) can be specified if you want the service to check more frequently than once per hour. A value of 0 (zero) does not check for stale generations.

Configuration Property	Value
`dataDirectory`	Specifies the location where the Record Store's data files are stored. The value of the `dataDirectory` property can either be an absolute path or a path relative to the directory where the Record Store is running. If not specified, the value defaults to `<install path>\IAS\<version>\workspace\state\<Record store name>`. This directory is created when a Record Store instance is created if it does not already exist. The Record Store service validates that the `dataDirectory` property specifies a directory for which the user running the Endeca IAS Service has write permissions.
`duplicateRecordCompressionEnabled`	Specifies whether to store new versions of records whose *change properties* have not changed. Enabling this feature improves Record Store performance and decreases Record Store disk space. The `duplicateRecordCompressionEnabled` property takes a Boolean value: `true` does not store duplicate copies of records, and `false` stores duplicate copies of records. If not specified, the value defaults to `false`.
`generationRetentionTime`	Specifies how long (in hours) a record generation should remain in a Record Store instance before it is considered a stale generation. The next time the `cleanerInterval` value is reached, then stale generations are deleted from the Record Store. However, the Record Store does not remove the most recent generation even if it exceeds the value of `generationRetentionTime`, and the Record Store does not remove the last generation specified with the `set-last-read-generation` task for a client ID. Fractional values (like `0.1`) can be specified if you want a generation to be maintained for less than an hour. If not specified, `generationRetentionTime` defaults to `48` hours (two days). The Record Store validates that the `generationRetentionTime` property is greater than or equal to `0`. If set to `0`, the Record Store instance only stores the latest (single) generation after the cleanup.

Configuration Property	Value
`idPropertyName`	Specifies the source property from which the record ID is derived. This value must be a non-empty string. If not specified, it defaults to `Endeca.Id`. When selecting a record property as the `idPropertyName`, you should choose a property that is present on every record and whose value is unique to each record. That is, all records (except those tagged with Delete All) must have a single unique non-null value for this property. The uniqueness of the property value is important because if two records have the same `idPropertyName` property value, the second record that is processed overwrites the first one in the Record Store. The Record Store validates that the `idPropertyName` property is a non-empty string. If this property is not correctly set, `RecordStore.setConfiguration()` throws a `RecordStoreConfigurationException`.
`ignoreInvalidRecords`	Specifies how invalid records are handled. Invalid records are records with missing IDs (either the `idPropertyName` property is missing or it has a null value) or with invalid action types for the `Endeca.Action` property. The `ignoreInvalidRecords` property takes a Boolean value: • If set to `true`, invalid records are ignored and a warning message is logged. The READ_WRITE operation for the records continues. • If set to `false`, an invalid record throws an exception and stops the process. In either case, invalid records are not added to the Record Store. If not specified, the value defaults to `true`. During the development stage of your Record Store application, you may want to set the `ignoreInvalidRecords` property to `false` so that an `InvalidRecordFault` exception is thrown whenever an invalid record is processed. This allows you to immediately see if your source records have the appropriate properties. Once you go into production, you can change the property to `true` and monitor the logs for warning messages about invalid records.

Configuration Property	Value
`indexWriteFlushInterval`	Oracle Internal Use. Oracle does not recommend modifying this property. This value must greater than `0`. The Record Store validates that the `indexWriteFlushInterval` property is greater than `0`. If this property is not correctly set, `RecordStore.setConfiguration()` throws a `RecordStoreConfigurationException`.
`jdbmSettings`	Oracle Internal Use. Oracle does not recommend modifying this property.
`maxDataFileSize`	Oracle Internal Use. Oracle does not recommend modifying this property. This value must be greater than `0`. The default value is 2 GB. The Record Store validates that the `maxDataFileSize` property is greater than `0`. If this property is not correctly set, `RecordStore.setConfiguration()` throws a `RecordStoreConfigurationException`.
`recordCompressionEnabled`	Specifies whether records are stored on disk in a compressed format. The `recordCompressionEnabled` property takes a Boolean value: • If set to `true`, records are stored on disk in a compressed format. • If set to `false`, records are stored on disk in an uncompressed format. If not specified, the value defaults to `false`.

Example of a configuration file for a Record Store instance

A sample configuration file is shown here:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<recordStoreConfiguration xmlns="http://recordstore.eidi.endeca.com/">
    <btreePageSize>100</btreePageSize>
    <changePropertyNames/>
    <cleanerInterval>1.0</cleanerInterval>
    <dataDirectory>C:\Oracle\Endeca\IAS\workspace\state\RS1\data</dataDirectory>
    <duplicateRecordCompressionEnabled>false</duplicateRecordCompressionEnabled>
    <generationRetentionTime>168.0</generationRetentionTime>
    <idPropertyName>Endeca.Id</idPropertyName>
    <ignoreInvalidRecords>false</ignoreInvalidRecords>
    <indexWriteFlushInterval>50000</indexWriteFlushInterval>
    <jdbmSettings/>
    <maxDataFileSize>2147483647</maxDataFileSize>
    <recordCompressionEnabled>false</recordCompressionEnabled>
</recordStoreConfiguration>
```

Change properties and new Record Store instances

When IAS creates a new Record Store instance for a crawl, it uses all record properties as change properties. In other words, IAS evaluates all properties on a record to determine if the record is different from the previous generation's record.

If desired, you can restrict the number of properties that IAS evaluates by configuring the `changePropertyNames` property. This improves performance by allowing IAS to check a sub-set of record properties when determining if a record has changed. In addition, IAS can check only properties that are meaningful to the data. For example, you can set a change property to check a file version number rather than a property for fetch time or trivial information.

Deleting stale generations of records

This topic provides guidance about how to delete stale generations of records by appropriately setting the `generationRetentionTime` property.

As a general rule, the value of `generationRetentionTime` should be greater than the sum of the following:

- The time between the start of two write operations to a Record Store instance.
- The time between the start of two delta read operations from a Record Store instance.
- Time for a margin of safety (For example, this includes time to revert to an earlier generation, fix any issues in the data, and re-crawl the data.)

For example, suppose a crawl, which writes to a Record Store, takes a few hours to run and runs once a day: the time between the start of two write operations is 24 hours. Next, suppose you run an Integrator graph once a day so the time between reads of the Record Store is 24 hours. Last, suppose you want to be able to revert to data up to three days old. You want a margin of safety of 72 hours. This means the value of `generationRetentionTime` should be at least 120. In this scenario, a value of 120 ensures there are two generations in a Record Store instance.

Note: The Record Store applies a read-lock to the generation being read. If a generation with a read-lock exceeds the `generationRetentionTime` value, the generation is not deleted until the read is complete and the read-lock is released.

Disabling automatic management of a Record Store instance

If you want maintain a Record Store instance separately from its associated crawl configuration, you can disable automatic management of the Record Store instance by the IAS Server and move a Record Store instance to another host as necessary. This may be useful if storage space for a Record Store instance is a concern and in some migration scenarios.

The `isManaged` property in a crawl configuration determines whether a Record Store instance is created or deleted at the same time as its associated crawl configuration. The `isManaged` property has a value of `true` by default. This means that:

- A Record Store instance is automatically created when you create a crawl. The name of a new Record Store instance corresponds directly to the crawl name.
- The associated Record Store instance is automatically deleted when you delete a crawl.

If you disable the `isManaged` property by setting it to `false`, a Record Store instance is not created when you create the crawl configuration. You must create the Record Store instance manually, or configure the crawl to send output to a file. Likewise, any Record Store instance that you create for a crawl configuration is not deleted when you delete the crawl configuration.

To disable automatic management of a Record Store instance:

1. From a command prompt, run the `getCrawl` task of `ias-cmd`. Use the `-f <arg>` flag to specify the name of the XML file to write the configuration to.
   For example, you might specify `-f configuration.xml`.
2. In the `configuration.xml` file for the crawl configuration, set the `isManaged` property to `false` as shown in the following example:

```
....
  <crawlConfig>
  ....
   <outputConfig>
    <moduleId>
      <id>Record Store</id>
    </moduleId>
    <moduleProperties>
     <moduleProperty>
      <key>isManaged</key>
      <value>false</value>
     </moduleProperty>
    </moduleProperties>
   </outputConfig>
....
  </crawlConfig>
....
```

3. Save and close the crawl configuration file.
4. Run the `updateCrawls` task of `ias-cmd` and pass in the crawl configuration file. For example:

```
updateCrawls -f configuration.xml
```

Performance considerations when using a Record Store instance

When reading or writing large numbers of records, some `READ_WRITE` operations can take long periods of time. Read operations generally take longer than write operations for similar size record sets, and the transaction time of a `READ_WRITE` operation grows approximately linearly as the number of records grows and the size of the records grows. For this reason, delta updates are generally faster than baseline updates.

If reading or writing operations cause performance concerns, there are several changes you can make:

- Use fast-access drives and RAID with striping on machines that host IAS record stores. This improves disk I/O for better overall record store performance.
- Reduce the `generationRetentionTime` setting so that fewer generations of records are stored. For details, see *Deleting stale generations of records on page 74*.
- Split a crawl into multiple crawls that use multiple Record Store instances.

Chapter 5

Running a Crawl

This section provides information about running a crawl using the IAS Server.

Running a crawl

You can run a crawl from the IAS Server Command-line Utility or programmatically from the IAS Server API.

Crawling from the IAS Server Command-line Utility

You can start and stop a crawl from the IAS Server Command-line Utility by running either the `startCrawl` or `stopCrawl` tasks. For details, see the chapter in this guide on the IAS Server Command-line Utility.

Crawling programmatically from the IAS Server API

You can start and stop a crawl by calling either the `IasCrawler.startCrawl()` method or the `IasCrawler.stopCrawl()` from an application. For details, see the *Endeca IAS API Guide*.

Order of execution in a crawl configuration

A crawl configuration specifies settings and processing instructions for a crawl. When you start a crawl, IAS Server executes the instructions in the following order: `sourceConfig`, `textExtractionConfig`, `manipulatorConfig`, and `outputConfig`. This topic provides additional detail about execution order.

When IAS Server starts a crawl, the following happens:

1. IAS Server crawls files and folders according to the seeds and settings in `sourceConfig`, and IAS Server creates an Endeca record for each file and folder crawled.
2. If `textExtractionConfig` is enabled and contains document conversion settings, then IAS Server performs document conversion and stores the converted text as a property on the Endeca record.

3. If one or more `manipulatorConfig` elements are present, IAS Server passes the record to each manipulator for processing according to its `manipulatorConfig` settings. Manipulators execute in the order in which they are nested within `manipulatorConfigs`.
4. IAS Server then writes the record to a Record Store instance (or an output file) according to the settings in according to `outputConfig`.

This processing continues until all files and folders are crawled and all records are processed. In this way, Endeca records are propagated through a crawl configuration.

Full and incremental crawling modes

The IAS Server starts a crawl in one of two modes:

- `full` mode, in which all content is processed.
- `incremental` mode, in which only new, modified, or deleted content is processed.

Crawling in full mode

Crawling in *full* mode means that IAS processes all the content in a data source according to the filtering criteria you specify. As part of crawling a data source, IAS creates metadata information and stores it in a crawl history. This history includes the ID of each record and information about all properties on the record.

Crawling in incremental mode

Crawling in *incremental* mode means that IAS processes only that content whose metadata information, stored in the crawl history, has changed since the last crawl. Specifically, IAS checks all properties on the record to see if any have changed. If any properties have changed, the IAS Server crawls the content again. This is true in cases where IAS is calculating the incremental difference. An extension developer, using the IAS Extension API, may choose to calculate incremental changes in a data source extension.

IAS automatically determines which crawling mode is necessary. By default, IAS attempts to crawl in incremental mode. If necessary, IAS switches to crawling in full mode, if a crawl's configuration has `unavailableIncrementalSwitchesToFullCrawl` set to `true`, and any of the following conditions are true:

- A data source has not been crawled before, which means no crawl history exists.
- A Record Store instance does not contain at least one record generation. (This applies to cases where the IAS Server is configured to output to a Record Store instance rather than a file on disk.)
- Seeds have been removed from the crawl configuration (adding seeds does not require crawling in full mode).
- The document conversion setting has changed.
- Folder filters or file filters have been added, modified, or removed in the data source configuration.
- Repository properties have been changed, such as the **Gather native properties** option for file system data sources.

If `unavailableIncrementalSwitchesToFullCrawl` is set to `false` and any of the above conditions are true, the crawl fails and throw and exception.

This switch from incremental to full mode can occur no matter how you run a crawl (using the IAS Server API or the IAS Server Command-line Utility).

After you crawl a data source using `ias-cmd` or the API, a status message is returned indicating whether a full or incremental crawl ran.

Crawls and archive files

File system and CMS crawls can process archive files.

Archive expansion is disabled by default. To enable the feature, you must set `expandArchives` to `true` in the crawl configuration file.

Archive expansion means that an Endeca record is created for each archived entry and its properties are populated. Text is extracted if the document conversion option is enabled. Note that native file properties are not gathered for archived entries even if that option is enabled for file system crawls. However, file and CMS permissions of the archive file are propagated to the archive entries.

Archive file support

An archive file is one that holds one or more archived entries (files or directories) within it. Two examples of archives are ZIP files and UNIX TAR files.

The IAS identifies archives by their file extensions for file system crawls, or mime types for CMS crawls. The following archive types are supported in file system crawls:

- JAR files (.jar extension)
- TAR files (.tar extension)
- GZIP-compressed Tar files (.tar.gz and .tgz extensions)
- ZIP files (.zip extension)

Support for ZIP files

ZIP files are supported as follows:

- ZIP files can have either no compression or the standard Deflate compression algorithm. ZIP files that use a compression scheme other than the Deflate algorithm are not treated as ZIP files. In this case, one record is created for the file, with the `Endeca.File.IsArchive` property set to `false`.
- There is no support for ZIP files with password-protected entries. ZIP files that contain password-protected entries are not fully processed. The actual behavior depends on the form of password protection:
  - If using the AES-128 or AES-256 forms of password encryption, the file is not marked as a ZIP file. One record is created for the file, with the `Endeca.File.IsArchive` property set to `false`.
  - If using the ZipCrypto password protection, the ZIP is recognized, and each entry that is encountered in order that is not password-protected will have a record created for it. Once a password-protected entry is encountered, the processing on the ZIP stops, and no further records are created.
  - For a number of ZIP utilities, directory entries are not password-protected (so that only the files are encrypted), and that directory entries are often put at the beginning of a ZIP. One record is created for the file, with the `Endeca.File.IsArchive` property set to `true`, and additional records are created for those (directories) that are not encrypted.

- There is no support for entries that are split across multiple ZIP files. Splitting a file over multiple ZIP files results in two kinds of ZIP files: those that store the partial data for the underlying file and and a "last" one that also stores the entry information. Different tools use different naming conventions, so sometimes the partials have a .zip extension and sometimes they do not. However, the last file will be a .zip file. These files are handled as follows:
  - The partial files will not be recognized as ZIP files. One record is created for the file, with the `Endeca.File.IsArchive` property set to `false`.
  - The last file will be recognized as a ZIP file, but its entry will be unreadable. One record is created for the file, with the `Endeca.File.IsArchive` property set to `true`.

When a Zip file is not treated as a valid ZIP file for any reason, the log file will contain a warning that the ZIP file in question contains an "invalid CEN header", and the record generated for the ZIP file will not indicate that it is an archive.

Note: JAR files are handled the same way as ZIP files. Therefore, any caveats that apply to ZIP files also apply to JAR files as well.

Support for Tar files

The supported Tar formats are the following:

- GNU Tar 1.13
- GNU Tar 1.12 or earlier
- UNIX V7
- POSIX.1-1988 (original USTAR format)
- Any of the above formats, compressed with GZip

Any format that is not listed above is considered an unsupported format. For example, the POSIX.1-2001 format is explicity not supported.

The IAS Server processes Tar files as follows:

- For supported formats, each entry in the Tar file is extracted and written as a record.
- For POSIX.1-2001 Tars, the entries are not extracted and a message is written to the log indicating that the format is not supported.
- For corrupted Tar entries:
  - If the first entry is corrupted, the entire Tar will not be extracted. Instead, it will be treated as any other non-archive file.
  - If any later entry is corrupted, the occurrence of the bad entry is logged. All prior entries are extracted and written as records to the output file.

How archive files are handled

The following is a detailed view of how the IAS Server handles archive files:

- An Endeca record is created for the archive file itself. This record has the `Endeca.File.IsArchive` property set to `true`.
- In addition to the top-level documents (files or directories), nested archive files are also processed.

- Document conversion (if enabled) is performed on all files within the archive, in accordance with document conversion filtering.
- A separate Endeca record is created for each document (including nested archives) found in the archive. The record is processed as follows:
  - The record has the `Endeca.File.IsInArchive` property set to `true`. In addition, the `Endeca.File.SourceArchive` and `Endeca.File.PathWithinSourceArchive` properties are added with a reference to the parent archive.
  - The filtering behavior works the same for archived files and directories (that is, files and directories in an archive) as it does for non-archived files and directories.
  - For records from either file system or CMS crawls, the record Id is a concatenation of the `Endeca.File.SourceArchiveId` property and the `Endeca.File.PathWithinSourceArchive` property:
    - For file system records, the `Endeca.FileSystem.Path` property is the record Id. This property is a canonical string pointing to the file within the archive, and follows this format:
      ```
      /path/to/archive//path/to/archivedfile
      ```
    - For CMS records, the `Endeca.Id` property is the record Id. This property is a canonical string pointing to the file within the archive, and follows this format:
      ```
      reposId:itemId[:optionalContentPieceId]//path/to/archivedfile
      ```

Note:

- Double delimiters represent the boundaries of the archive.
- Path delimiters for the value of the `PathWithinSourceArchive` property appear as forward slashes (they are platform-independent).
- Path delimiters for the value of the `Endeca.FileSystem.Path` property are platform-dependent, so in the case of Windows files, path delimeters on this property appear as backslashes. For example:
  ```
  C:\path\to\archive\\path\to\archivedfile
  ```

In the case of nested archives, the `Endeca.File.PathWithinSourceArchive` property takes the following format:

```
//path/to/nested/archive//path/within/nested/archive
```

- While the properties of archived entries are obtained in an Endeca record, the entries themselves are not physically extracted from the archive (that is, no new files are permanently saved to disk).
- If an archive has entries with identical names, the first entry that is processed is kept (that is, an Endeca record is created for it) and the duplicate entry is ignored.
- Seeds are restricted to actual files or directories or entries. That is, seeds cannot point to archived files or directories.

The above behavior is the default for all archives crawled. To avoid processing archives, disable the Expand archives option for the crawl.

About writing records to a Record Store instance

A Record Store instance is tightly integrated with the output produced by IAS Server. IAS Server writes the output for file system and CMS repository directly to a Record Store instance by default, instead of to a file on disk.

About the record output file

This topic describes record output files for full and incremental crawl modes.

You configure the attributes of an record output file from the IAS Server API or in a crawl configuration file that you provide to a command in the IAS Server Command-line Utility.

For example, you set the path of the output directory with the `outputDirectory` property in the API or path in the configuration file. If you do not specify an output directory, a default name of `output` is used for the `crawlID` sub-directory and it is located in the IAS Server's workspace directory.

Record output file

The prefix for the name of a crawl output file is set by the `outputPrefix` property (in the API) or key (in the configuration file). If you do not specify an output prefix, a default name of `CrawlerOutput` is used.

The full name of the output file also depends on two other configuration settings:

- The `outputXml` property. This specifies whether the output format is XML (with a file extension of `.xml`) or Binary (with a file extension of `.bin`).
- The `outputCompressed` property. This determines whether the output file is compressed. If compression is enabled, a `.gz` file extension is added to the `.xml` or `.bin` extension. No extension is added if compression is not enabled.

In addition to the output prefix described above, a second prefix is automatically added to the filename to distinguish which type of crawl was run:

- For full crawls, the `-FULL` suffix is added (e.g., `CrawlerOutput-FULL.xml`).
- For incremental crawls, the `-INCR` suffix is added (e.g., `CrawlerOutput-INCR.xml`).

The maximum size of a binary output file is 512 megabytes. If the maximum size is reached and more records need to be output, the crawler rolls the output into another output file. To distinguish rollover files, the `-sgmt000` prefix is added to the first file, `-sgmt001` is added to the second file, and so on, as shown in this example:

```
CrawlerOutput-FULL-sgmt000.bin.gz
CrawlerOutput-FULL-sgmt001.bin.gz
```

The maximum size of binary output files is not configurable. Note that unlike the binary format, if you choose XML, only one file is output, regardless of its size.

Archived output files

The first time that IAS Server crawls a data source, the output file is named as described in the previous section. For example, if you run a full crawl, the output filename might be `CrawlerOutput-FULL-`

`sgmt000.bin.gz`. If you then run a second crawl (for example, an incremental crawl), the IAS Server works as follows:

1. A directory named `archive` is created under the output directory.
2. The original `CrawlerOutput-FULL-sgmt000.bin.gz` file is moved to the `archive` directory and is renamed by adding a timestamp to the name; for example:
   ```
   CrawlerOutput-FULL-20071026140235-sgmt000.bin.gz
   ```
3. The output file from a second incremental run is named `CrawlerOutput-INCR-sgmt000.bin.gz` and is stored in the output directory.
4. For every subsequent crawl using the same output directory, steps 2 and 3 are repeated.

The timestamp format used for renaming is:

```
YYYYMMDDHHmmSS
```

where:

- YYYY is a four-digit year, such as 2009.
- MM is the month as a number (01-12), such as 10 for October.
- DD is the day of the month, such as 25 (for October 25th).
- HH is the hour of the day in a 24-hour format (00-23), such as 14 (for 2 p.m.).
- mm is the minute of the hour (00-59).
- SS is the second of the minute (00-59).

The timestamp format is not configurable.

Part III

IAS Command Line Utilities

Chapter 6

IAS Server Command-line Utility

This section describes how to run the tasks of the IAS Server Command-line Utility.

Overview of the IAS Server Command-line Utility

The IAS Server Command-line Utility creates, runs, and manages crawls.

The IAS Server Command-line Utility is a script named `ias-cmd.sh` (for Linux/UNIX systems) and `ias-cmd.bat` (for Windows) that you run from a command prompt. The scripts are in the `bin` directory.

Help options

The IAS Server Command-line Utility has two help options that display the usage syntax. The `--help` option displays a summary of the tasks. The `--help-detail` option displays detailed usage information for all the tasks. For example:

```
C:Oracle\Endeca\IAS\3.0.0\bin>ias-cmd.bat --help
usage: ias-cmd <task-name> [options]
[Inspecting Installed Modules]
  getAllModuleSpecs
  getModuleSpec
  listModules
[Managing Crawls]
  createCrawls
  deleteCrawl
  getAllCrawls
  getCrawl
  getCrawlIncrementalSupport
  listCrawls
  startCrawl
  stopCrawl
  updateCrawls
[Viewing Crawl Status and Results]
  getAllCrawlMetrics
  getCrawlMetrics
  getCrawlStatus

For detailed usage information including task options, use --help-detail
For detailed usage information for individual task options, use <task-name>
```

```
--help
```

Command-line options

The command syntax for executing the tasks is:

```
ias-cmd task-name [options]
```

The *task-name* argument is the task to be performed by the utility, such as the `createCrawls` task. The task options vary, depending on the task. However, these options can be used with any task:

- `-h` (or `--host`) specifies the host name of the machine on which the IAS Service is running. If the flag is omitted, it defaults to the value of the `com.endeca.eidi.ias.server.host` property in `<install path>\IAS\workspace\conf\commandline.properties`. If the property is not set, the value then defaults to `localhost` as the host name.
- `-p` (or `--port`) specifies the port on which IAS Service is listening. If the flag is omitted, it defaults to the value of the `com.endeca.eidi.ias.server.port` property in `workspace\IAS\conf\commandline.properties`. If the property is not set, the value then defaults to `8510` as the port number.
- `-l` (or `--isPortSsl`) specifies whether to communicate with the service using an HTTPS connection. A value of `true` uses HTTPS and treats the `com.endeca.eidi.ias.server.port` property as an SSL port. A value of `false` uses HTTP and treats `com.endeca.eidi.ias.server.port` as a non-SSL port. Specify `false` if you enabled redirects from a non-SSL port to an SSL port. Optional.

Host and port settings

You first specify the host and port settings for the Endeca IAS Service as part of the installation wizard. That host runs the IAS Server, the Component Instance Manager, and all Record Store instances. The installation wizard then writes the host (`com.endeca.eidi.ias.server.host`) and port (`com.endeca.eidi.ias.server.port`) settings as properties in `commandline.properties`. All of the IAS command-line utilities use these settings as default values if you omit the `-h` and `-p` flags when executing any tasks.

Setting the bin directory in the PATH environment variable

Although not required, it is recommended that you set the path of the `bin` directory in your system's PATH environment variable. This allows you to run the IAS Server Command-line Utility script from any location.

About error handling

- If desired, you can re-configure the default logging settings in `<install path>\IAS\workspace\conf\ias-cmd.log4j.properties`.
- Errors print to standard error, unless you redirect `stderr` to a file instead.
- Errors of mis-configured command-line tasks or incorrect input parameters are written to standard out.

About IAS capabilities

The Integrator Acquisition System provides a list of capabilities that describe whether a crawl type or manipulator supports optional IAS features. For example, if a crawl or manipulator has the Supports Incrementals capability, then it can run in an incremental crawl.

You get the capabilities for a crawl or manipulator by running the `listModules` task or the `getModuleSpec` task of `ias-cmd`.

The list of IAS capabilities available to a crawl or manipulator includes the following:

- Binary Content Accessible via FileSystem - Indicates that the crawl supports local caching for files accessible from a file system. This capability does not apply to manipulators.
- Data Source Filter - Indicates that the crawl supports filter configuration. This capability does not apply to manipulators.
- Has Binary Content - Indicates that the crawl supports document conversion. This capability does not apply to manipulators.
- Expand Archives - indicates that the crawl supports archive expansion as part of a crawl. This capability does not apply to manipulators.
- Supports Incrementals - Indicates that the manipulator can run as part of an incremental crawl. This capability does not apply to crawls.

Saving passwords in a crawl configuration file

Although crawls can be configured with passwords, a crawl configuration file retrieved by the `getCrawl` or `getAllCrawls` does not retrieve passwords.

There are two ways to specify a password for a crawl:

- You can specify a password when prompted by the `createCrawls` task of `ias-cmd`.
- You can save the password in a crawl configuration file.

Note: The `updateCrawls` task of `ias-cmd` does not prompt for a password because IAS Server stores the password during the create process, and the `updateCrawls` task uses the stored password.

Recall that passwords are specified in crawls with the `password` configuration property. However, in cases where a plug-in developer creates a data source or manipulator with a password configuration property, the property may have any name the plug-in developer chooses. (In this situation, the plug-in developer specifies a password configuration property by adding the `isPassword=true` attribute in the property's annotation.)

To save a password in a crawl configuration:

1. In a text editor, open the crawl configuration file and locate the `<configuration>` element for the given crawl and within `<configuration>` locate the `<sourceConfig>` element.
2. Within `<sourceConfig>`, locate the `<moduleProperty>` element that specifies the password configuration property.
   - For default crawls, this is the `<moduleProperty>` with `<key>password</key>`.

- For data sources or manipulators created by a plug-in developer, you can locate the password configuration property by running the `getModuleSpec` and looking for the property that has `*Password: true`.

For example:

```
<moduleProperty>
   <key>password</key>
</moduleProperty>
```

3. Directly below the `<key> ...</key>` line, enter `<value>` followed by a value you wish to set as the password, and then the closing `</value>` tag.

   For example:

```
<moduleProperty>
   <key>password</key>
   <value>p@ssw0rd</value>
</moduleProperty>
```

4. Save and close the configuration file.
5. Specify this configuration file with the `-f` option of the `createCrawls` task.

Inspecting installed modules

The following `ias-cmd` tasks return information about the modules you have installed.

Getting the specifications of all modules

The `getAllModuleSpecs` task retrieves all module specifications. A module specification includes the configuration properties, capabilities, and `moduleInfo` of a particular module.

The syntax for this task is:

```
ias-cmd getAllModuleSpecs [-h HostName] [-l true|false] [-p PortNumber] [-t ModuleType]
```

Where:

- `-h` (or `--host`) specifies the host where the Endeca IAS Service is running. If the flag is omitted, the default is the value of the `com.endeca.eidi.ias.server.host` property in `commandline.properties`. If the property is not set, the value then defaults to `localhost`. Optional.
- `-l` (or `--isPortSsl`) specifies whether to communicate with the service using an HTTPS connection. A value of `true` uses HTTPS and treats the `com.endeca.eidi.ias.server.port` property as an SSL port. A value of `false` uses HTTP and treats `com.endeca.eidi.ias.server.port` as a non-SSL port. Specify `false` if you enabled redirects from a non-SSL port to an SSL port. Optional.
- `-p` (or `--port`) specifies the port of the Endeca IAS Service. If the flag is omitted, the default is the value of the `com.endeca.eidi.ias.server.port` property in `commandline.properties`. If the property is not set, the value then defaults to `8510`. Optional.
- `-t` (or `--type`) specifies the type of module to list. If unspecified, the task returns the specifications of all modules. A value of `SOURCE` returns the specifications of all data sources. A value of `MANIPULATOR` returns the specifications of all manipulators. Optional.

To get the specifications of all modules:

1. Start a command prompt and navigate to `<install path>\IAS\<version>\bin` (for Windows), or `<install path>/IAS/<version>/bin` (for UNIX).
2. Type `ias-cmd.bat` (for Windows), or `ias-cmd.sh` (for UNIX) and specify the `getAllModuleSpecs` task.

Note: This task name is case sensitive.

Example of getting the specifications of all modules

```
ias-cmd getAllModuleSpecs
Documentum Content Server
=========================
[Module Information]
 *Id: Documentum Content Server
 *Type: SOURCE
 *Description: No description available for Documentum Content Server
 *Capabilities:
   *Data Source Filter
   *Has Binary Content
   *Expand Archives

[Documentum Content Server Configuration Properties]
Group: Credentials
------------------
User name:
 *Name: username
 *Type: {http://www.w3.org/2001/XMLSchema}string
 *Required: true
 *Max Length: 256
 *Description: The name of the user used to log on to the repository
 *Multiple Values: false
 *Multiple Lines: false
 *Password: false
 *Always Show: true

Password:
 *Name: password
 *Type: {http://www.w3.org/2001/XMLSchema}string
 *Required: true
 *Max Length: 256
 *Description: The password used to log on to the repository
 *Multiple Values: false
 *Multiple Lines: false
 *Password: true
 *Always Show: true

Domain:
 *Name: domain
 *Type: {http://www.w3.org/2001/XMLSchema}string
 *Required: false
 *Max Length: 256
 *Description: The domain of the user used to log on to the repository
 *Multiple Values: false
 *Multiple Lines: false
 *Password: false
 *Always Show: true

Group: Repository Configurations
--------------------------------
 *Help Link: <?>
Docbase:
 *Name: docbase
```

```
 *Type: {http://www.w3.org/2001/XMLSchema}string
 *Required: true
 *Possible Values:
   *Label: dctm65 / Value: dctm65
 *Description: The name of the Documentum Docbase.
 *Multiple Values: false
 *Multiple Lines: false
 *Password: false
 *Always Show: false

Group: Seeds
------------
 *Help Link: <?>
Seeds:
 *Name: seeds
 *Type: {http://www.w3.org/2001/XMLSchema}string
 *Required: false
 *Max Length: 16336
 *Multiple Values: true
 *Multiple Lines: false
 *Password: false
 *Always Show: true

...
```

Getting the specification of a module

The `getModuleSpec` task retrieves the specification of a particular module. A module specification includes the configuration properties, capabilities, and `moduleInfo` of a particular module.

The syntax for this task is:

```
ias-cmd getModuleSpec -id ModuleId [-h HostName]
[-p PortNumber] [-l true|false]
```

Where:

- `-id` (or `--module_id`) specifies the ID of a module that you have installed into IAS. For example, a CMS data source may be called `Lotus Notes` or `Microsoft Sharepoint`.
- `-h` (or `--host`) specifies the host where the Endeca IAS Service is running. If the flag is omitted, the default is the value of the `com.endeca.eidi.ias.server.host` property in `commandline.properties`. If the property is not set, the value then defaults to `localhost`. Optional.
- `-p` (or `--port`) specifies the port of the Endeca IAS Service. If the flag is omitted, the default is the value of the `com.endeca.eidi.ias.server.port` property in `commandline.properties`. If the property is not set, the value then defaults to `8510`. Optional.
- `-l` (or `--isPortSsl`) specifies whether to communicate with the service using an HTTPS connection. A value of `true` uses HTTPS and treats the `com.endeca.eidi.ias.server.port` property as an SSL port. A value of `false` uses HTTP and treats `com.endeca.eidi.ias.server.port` as a non-SSL port. Specify `false` if you enabled redirects from a non-SSL port to an SSL port. Optional.

If necessary you can first run the `listModules` task to list the modules that you have installed.

To get the specification of a module:

1. Start a command prompt and navigate to `<`*`install path`*`>\IAS\<version>\bin` (for Windows), or `<`*`install path`*`>/IAS/<version>/bin` (for UNIX).

2. Type `ias-cmd.bat` (for Windows), or `ias-cmd.sh` (for UNIX) and specify the `getModuleSpec` task with the id of the module for which to retrieve the specification.

Note: This task name is case sensitive.

Example of getting the specification of a module

```
ias-cmd.bat getModuleSpec -id "File System"
File System
===========
[Module Information]
 *Id: File System
 *Type: SOURCE
 *Description: The File System data source crawls Windows and UNIX file systems.

 *Capabilities:
   *Binary Content Accessible via FileSystem
   *Data Source Filter
   *Has Binary Content
   *Expand Archives

[File System Configuration Properties]
Group: Seeds
------------
 *Help Link: <?>
Seeds:
 *Name: seeds
 *Type: {http://www.w3.org/2001/XMLSchema}string
 *Required: true
 *Max Length: 255
 *Multiple Values: true
 *Multiple Lines: false
 *Password: false
 *Always Show: true

Group:
-------
Gather Native File Properties:
 *Name: gatherNativeFileProperties
 *Type: {http://www.w3.org/2001/XMLSchema}boolean
 *Required: false
 *Description: Gather Native File Properties
 *Multiple Values: false
 *Multiple Lines: false
 *Password: false
 *Always Show: false

Expand Archives:
 *Name: expandArchives
 *Type: {http://www.w3.org/2001/XMLSchema}boolean
 *Required: false
 *Description: Expand Archives
 *Multiple Values: false
 *Multiple Lines: false
 *Password: false
 *Always Show: false
```

Listing modules

The `listModules` task lists modules you can include in a crawl. Modules include CMS crawls and any custom data sources or manipulators.

The syntax for this task is:

```
ias-cmd listModules [-t ModuleType] [-h HostName] [-p PortNumber] [-l true|false]
```

Where:

- `-t` (or `--type`) specifies the type of module to list. If unspecified, the task returns all modules. A value of `SOURCE` returns a list of all data sources enabled on the IAS Server. A value of `MANIPULATOR` returns a list of all manipulators installed on the IAS Server. Optional.
- `-h` (or `--host`) specifies the host where the Endeca IAS Service is running. If the flag is omitted, the default is the value of the `com.endeca.eidi.ias.server.host` property in `commandline.properties`. If the property is not set, the value then defaults to `localhost`. Optional.
- `-p` (or `--port`) specifies the port of the Endeca IAS Service. If the flag is omitted, the default is the value of the `com.endeca.eidi.ias.server.port` property in `commandline.properties`. If the property is not set, the value then defaults to `8510`. Optional.
- `-l` (or `--isPortSsl`) specifies whether to communicate with the service using an HTTPS connection. A value of `true` uses HTTPS and treats the `com.endeca.eidi.ias.server.port` property as an SSL port. A value of `false` uses HTTP and treats `com.endeca.eidi.ias.server.port` as a non-SSL port. Specify `false` if you enabled redirects from a non-SSL port to an SSL port. Optional.

To list modules:

1. Start a command prompt and navigate to `<install path>\IAS\<version>\bin` (for Windows), or `<install path>/IAS/<version>/bin` (for UNIX).
2. Type `ias-cmd.bat` (for Windows), or `ias-cmd.sh` (for UNIX) and specify the `listModules` task.

Note: This task name is case sensitive.

Example of listing modules

```
ias-cmd listModules -t SOURCE
Documentum Content Server
 *Id: Documentum Content Server
 *Type: SOURCE
 *Description: No description available for Documentum Content Server
 *Capabilities:
   *Data Source Filter
   *Has Binary Content
   *Expand Archives

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
```

Managing crawls

The following `ias-cmd` tasks manage crawls.

Creating crawls

The `createCrawls` task creates and stores named crawls.

The syntax for this task is:

```
ias-cmd createCrawls -f CrawlConfig.xml [-h HostName] [-p PortNumber] [-l true|false]
```

Where:

- `-f` (or `--file_name`) specifies the pathname of the input XML file containing the crawl configuration(s). Required.
- `-h` (or `--host`) specifies the host where the Endeca IAS Service is running. If the flag is omitted, the default is the value of the `com.endeca.eidi.ias.server.host` property in `commandline.properties`. If the property is not set, the value then defaults to `localhost`. Optional.
- `-p` (or `--port`) specifies the port of the Endeca IAS Service. If the flag is omitted, the default is the value of the `com.endeca.eidi.ias.server.port` property in `commandline.properties`, in *<install path>*`\IAS\workspace\conf` (on Windows) or *<install path>*`/IAS/workspace/conf` (on UNIX). If the property is not set, the value then defaults to `8510`. Optional.
- `-l` (or `--isPortSsl`) specifies whether to communicate with the service using an HTTPS connection. A value of `true` uses HTTPS and treats the `com.endeca.eidi.ias.server.port` property as an SSL port. A value of `false` uses HTTP and treats `com.endeca.eidi.ias.server.port` as a non-SSL port. Specify `false` if you enabled redirects from a non-SSL port to an SSL port. Optional.

Note:

- If you are running `createCrawls` as part of migrating from a previous version of IAS to the current version, the `createCrawls` task handles updating all aspects of the crawl configuration file.
- If conflicts arise when running the `createCrawls` task (such as multiple crawl configurations occuring with the same `crawlId`), the utility prompts you to either ignore the listed conflicts and continue creating the rest of the crawls, or to abort the task. If a crawl cannot be created, the IAS Server logs an error and ignores that crawl.
- When the IAS Server Command-line Utility loads a crawl configuration that contains an empty password property, the user is prompted for a password. If a password is entered incorrectly, the crawl is not saved.
- You may add a password to the crawl configuration and update IAS Server with this modified configuration. Or, you may enter the password when prompted by running the task. The password is saved only on the server running the Endeca IAS Service.

To create crawls:

1. Start a command prompt and navigate to *<install path>*`\IAS\<version>\bin` (for Windows), or *<install path>*`/IAS/<version>/bin` (for UNIX).

2. Type `ias-cmd.bat` (for Windows), or `ias-cmd.sh` (for UNIX) and specify `createCrawls` with the `-f` or `--file_name` flag, and the absolute path to the crawl configuration file.

Note: This task name is case sensitive.

Example of creating crawls

```
ias-cmd createCrawls -f C:\tmp\fileCrawlConfig.xml
Created crawl FileCrawl
```

Deleting a crawl

The `deleteCrawl` task deletes a crawl.

The syntax for this task is:

```
ias-cmd deleteCrawl -id CrawlName [-h HostName] [-p PortNumber] [-l true|false]
```

Where:

- `-id` (or `--crawl_id`) specifies the name of the crawl to be deleted. Required.
- `-h` (or `--host`) specifies the host where the Endeca IAS Service is running. If the flag is omitted, the default is the value of the `com.endeca.eidi.ias.server.host` property in `commandline.properties`. If the property is not set, the value then defaults to `localhost`. Optional.
- `-p` (or `--port`) specifies the port of the Endeca IAS Service. If the flag is omitted, the default is the value of the `com.endeca.eidi.ias.server.port` property in `commandline.properties`. If the property is not set, the value then defaults to `8510`. Optional.
- `-l` (or `--isPortSsl`) specifies whether to communicate with the service using an HTTPS connection. A value of `true` uses HTTPS and treats the `com.endeca.eidi.ias.server.port` property as an SSL port. A value of `false` uses HTTP and treats `com.endeca.eidi.ias.server.port` as a non-SSL port. Specify `false` if you enabled redirects from a non-SSL port to an SSL port. Optional.

1. Start a command prompt and navigate to `<install path>\IAS\<version>\bin` (for Windows), or `<install path>/IAS/<version>/bin` (for UNIX).
2. Type `ias-cmd.bat` (for Windows), or `ias-cmd.sh` (for UNIX) and specify the `deleteCrawl` task with the id of the crawl to be deleted.

Note: This task name is case sensitive.

Example of deleting a crawl

```
ias-cmd deleteCrawl -id FileCrawl
```

Getting all crawls

The `getAllCrawls` task retrieves all crawl configurations.

The syntax for this task is:

```
ias-cmd getAllCrawls [-f FileName.xml] [-h HostName] [-p PortNumber]
```

```
[-d] [-l true|false]
```

Where:

- `-f` (or `--file_name`) specifies the name of the XML file to write the configuration to. If omitted, the crawl configuration is sent to standard output. Optional.
- `-h` (or `--host`) specifies the host where the Endeca IAS Service is running. If the flag is omitted, the default is the value of the `com.endeca.eidi.ias.server.host` property in `commandline.properties`. If the property is not set, the value then defaults to `localhost`. Optional.
- `-p` (or `--port`) specifies the port of the Endeca IAS Service. If the flag is omitted, the default is the value of the `com.endeca.eidi.ias.server.port` property in `commandline.properties`. If the property is not set, the value then defaults to `8510`. Optional.
- `-d` (or `--fill_in_defaults`) specifies whether to populate the configuration file with the default values for unspecified properties. Optional.
- `-l` (or `--isPortSsl`) specifies whether to communicate with the service using an HTTPS connection. A value of `true` uses HTTPS and treats the `com.endeca.eidi.ias.server.port` property as an SSL port. A value of `false` uses HTTP and treats `com.endeca.eidi.ias.server.port` as a non-SSL port. Specify `false` if you enabled redirects from a non-SSL port to an SSL port. Optional.

Crawls are retrieved without password values if there are any configuration properties marked as `isPassword`.

To get all crawls:

1. Start a command prompt and navigate to `<install path>\IAS\<version>\bin` (for Windows), or `<install path>/IAS/<version>/bin` (for UNIX).
2. Type `ias-cmd.bat` (for Windows), or `ias-cmd.sh` (for UNIX) and specify the `getAllCrawls` task, optionally with the `-f` (or `--file`) flag and the name of the XML file to write the crawl configuration(s) to.

Note: This task name is case sensitive.

Example of getting all crawls

```
ias-cmd getAllCrawls
<?xml version="1.0" encoding="UTF-8"?>

<configurations xmlns="http://endeca.com/eidi/ias/2011-12">
  <crawlConfig>
    <crawlId>
      <id>FileCrawl</id>
    </crawlId>
    <unavailableIncrementalSwitchesToFullCrawl>true</unavailableIncrementalSwitchesToFullCrawl>
    <sourceConfig>
      <moduleId>
        <id>File System</id>
      </moduleId>
      <moduleProperties>
        <moduleProperty>
          <key>expandArchives</key>
          <value>false</value>
        </moduleProperty>
        <moduleProperty>
          <key>gatherNativeFileProperties</key>
          <value>true</value>
        </moduleProperty>
```

```
        <moduleProperty>
          <key>seeds</key>
          <value>C:\tmp\itldocset</value>
          <value>C:\tmp\iapdocset</value>
        </moduleProperty>
      </moduleProperties>
      <excludeFilters/>
      <includeFilters/>
    </sourceConfig>
    <textExtractionConfig>
      <enabled>true</enabled>
      <makeLocalCopy>false</makeLocalCopy>
    </textExtractionConfig>
    <manipulatorConfigs/>
    <outputConfig>
      <moduleId>
        <id>Record Store</id>
      </moduleId>
      <moduleProperties/>
    </outputConfig>
  </crawlConfig>
  <crawlConfig>
    <crawlId>
      <id>SecondFileCrawl</id>
    </crawlId>
    <sourceConfig>
      <moduleId>
        <id>File System</id>
      </moduleId>
      <moduleProperties>
        <moduleProperty>
          <key>expandArchives</key>
          <value>false</value>
        </moduleProperty>
        <moduleProperty>
          <key>gatherNativeFileProperties</key>
          <value>true</value>
        </moduleProperty>
        <moduleProperty>
          <key>seeds</key>
          <value>C:\tmp\mdexdocset</value>
        </moduleProperty>
      </moduleProperties>
      <excludeFilters/>
      <includeFilters/>
    </sourceConfig>
    <textExtractionConfig>
      <enabled>true</enabled>
      <makeLocalCopy>false</makeLocalCopy>
    </textExtractionConfig>
    </manipulatorConfigs>
    <outputConfig>
      <moduleId>
        <id>Record Store</id>
      </moduleId>
      <moduleProperties/>
    </outputConfig>
  </crawlConfig>
</configurations>
```

Getting a crawl

The `getCrawl` task retrieves a single crawl configuration.

The syntax for this task is:

```
ias-cmd getCrawl -id CrawlName [-f FileName.xml] [-h HostName]
[-p PortNumber] [-d] [-l true|false]
```

Where:

- `-id` (or `--crawl_id`) specifies the name of the crawl for which you want to retrieve the crawl configuration. Required.
- `-f` (or `--file_name`) specifies the XML output file to which you want to write the crawl configuration. Optional.
- `-h` (or `--host`) specifies the host where the Endeca IAS Service is running. If the flag is omitted, the default is the value of the `com.endeca.eidi.ias.server.host` property in `commandline.properties`. If the property is not set, the value then defaults to `localhost`. Optional.
- `-p` (or `--port`) specifies the port of the Endeca IAS Service. If the flag is omitted, the default is the value of the `com.endeca.eidi.ias.server.port` property in `commandline.properties`. If the property is not set, the value then defaults to `8510`. Optional.
- `-d` (or `--fill_in_defaults`) specifies whether to populate the configuration file with the default values for unspecified properties. Optional.
- `-l` (or `--isPortSsl`) specifies whether to communicate with the service using an HTTPS connection. A value of `true` uses HTTPS and treats the `com.endeca.eidi.ias.server.port` property as an SSL port. A value of `false` uses HTTP and treats `com.endeca.eidi.ias.server.port` as a non-SSL port. Specify `false` if you enabled redirects from a non-SSL port to an SSL port. Optional.

The XML input and output files resulting from the `getAllCrawls` and `createCrawls` operations are similar to those from `getCrawl`, except that `getAllCrawls` returns a series of `<crawlConfig>` elements because it pertains to multiple crawls.

Crawls are retrieved without password values if there are any configuration properties marked as `isPassword`.

To get a crawl:

1. Start a command prompt and navigate to `<`*`install path`*`>\IAS\<version>\bin` (for Windows), or `<`*`install path`*`>/IAS/<version>/bin` (for UNIX).
2. Type `ias-cmd.bat` (for Windows), or `ias-cmd.sh` (for UNIX) and specify the `getCrawl` task with the id of the crawl for which to retrieve the configuration.

   Note: This task name is case sensitive.

Example of getting a crawl

```
ias-cmd getCrawl -id FileCrawl
<?xml version="1.0" encoding="UTF-8"?>

<configurations xmlns="http://endeca.com/eidi/ias/2011-12">
  <crawlConfig>
    <crawlId>
      <id>FileCrawl</id>
    </crawlId>
    <unavailableIncrementalSwitchesToFullCrawl>true</unavailableIncrementalSwitchesToFullCrawl>
    <sourceConfig>
      <moduleId>
        <id>File System</id>
      </moduleId>
      <moduleProperties>
```

```
        <moduleProperty>
          <key>expandArchives</key>
          <value>false</value>
        </moduleProperty>
        <moduleProperty>
          <key>gatherNativeFileProperties</key>
          <value>true</value>
        </moduleProperty>
        <moduleProperty>
          <key>seeds</key>
          <value>C:\tmp\itldocset</value>
          <value>C:\tmp\iapdocset</value>
        </moduleProperty>
      </moduleProperties>
      <excludeFilters/>
      <includeFilters/>
    </sourceConfig>
    <textExtractionConfig>
      <enabled>true</enabled>
      <makeLocalCopy>false</makeLocalCopy>
    </textExtractionConfig>
    <manipulatorConfigs/>
    <outputConfig>
      <moduleId>
        <id>Record Store</id>
      </moduleId>
      <moduleProperties/>
    </outputConfig>
  </crawlConfig>
</configurations>
```

Getting the incremental support status of a crawl

The `getCrawlIncrementalSupport` task indicates whether a specified crawl configuration supports incremental crawling and also indicates which manipulators within the crawl configuration do not support incremental crawling.

The syntax for this task is:

```
ias-cmd getCrawlIncrementalSupport [-h HostName] -id CrawlName [-l true|false] [-p PortNumber]
```

Where:

- `-h` (or `--host`) specifies the host where the Endeca IAS Service is running. If the flag is omitted, the default is the value of the `com.endeca.eidi.ias.server.host` property in `commandline.properties`. If the property is not set, the value then defaults to `localhost`. Optional.
- `-id` (or `--crawl_id`) specifies the name of the crawl to retrieve incremental support status for. Required.
- `-l` (or `--isPortSsl`) specifies whether to communicate with the service using an HTTPS connection. A value of `true` uses HTTPS and treats the `com.endeca.eidi.ias.server.port` property as an SSL port. A value of `false` uses HTTP and treats `com.endeca.eidi.ias.server.port` as a non-SSL port. Specify `false` if you enabled redirects from a non-SSL port to an SSL port. Optional.
- `-p` (or `--port`) specifies the port of the Endeca IAS Service. If the flag is omitted, the default is the value of the `com.endeca.eidi.ias.server.port` property in `commandline.properties`. If the property is not set, the value then defaults to `8510`. Optional.

To get the incremental support status of an incremental crawl:

1. Start a command prompt and navigate to `<install path>\IAS\<version>\bin` (for Windows), or `<install path>/IAS/<version>/bin` (for UNIX).

2. Type `ias-cmd.bat` (for Windows), or `ias-cmd.sh` (for UNIX) and specify the `getCrawlIncrementalSupport` task with the id of the crawl.

Note: This task name is case sensitive.

Example of getting the support status of an incremental crawl

```
ias-cmd getCrawlIncrementalSupport -id Test
Incrementals Supported: yes
```

Listing crawls

The `listCrawls` task lists all crawls in the Endeca IAS Service.

The syntax for this task is:

```
ias-cmd listCrawls [-h HostName] [-p PortNumber] [-l true|false]
```

Where:

- `-h` (or `--host`) specifies the host where the Endeca IAS Service is running. If the flag is omitted, the default is the value of the `com.endeca.eidi.ias.server.host` property in `commandline.properties`. If the property is not set, the value then defaults to `localhost`. Optional.
- `-p` (or `--port`) specifies the port of the Endeca IAS Service. If the flag is omitted, the default is the value of the `com.endeca.eidi.ias.server.port` property in `commandline.properties`. If the property is not set, the value then defaults to `8510`. Optional.
- `-l` (or `--isPortSsl`) specifies whether to communicate with the service using an HTTPS connection. A value of `true` uses HTTPS and treats the `com.endeca.eidi.ias.server.port` property as an SSL port. A value of `false` uses HTTP and treats `com.endeca.eidi.ias.server.port` as a non-SSL port. Specify `false` if you enabled redirects from a non-SSL port to an SSL port. Optional.

To list crawls:

1. Start a command prompt and navigate to `<install path>\IAS\<version>\bin` (for Windows), or `<install path>/IAS/<version>/bin` (for UNIX).
2. Type `ias-cmd.bat` (for Windows), or `ias-cmd.sh` (for UNIX) and specify the `listCrawls` task.

Note: This task name is case sensitive.

Example of listing crawls

```
ias-cmd listCrawls
FileCrawl
FileCrawl2
```

Starting a crawl

The `startCrawl` task runs a crawl in whichever crawl mode is necessary.

By default, IAS runs incremental crawl, and switches to run a full crawl if any of the following conditions are true:

- A crawl has not been run before, which means no crawl history exists.
- A Record Store instance does not contain at least one record generation.
- Seeds have been removed from the crawl configuration (adding seeds does not require full crawl).
- The document conversion setting has changed.
- Filters have been added, modified, or removed in the crawl configuration.
- Repository properties have changed, such as the `username` property setting for CMS crawl.

In all other cases, the IAS Server starts a crawl in incremental mode. However, you may force full crawl by specifying the `-full` option.

The syntax for this task is:

```
ias-cmd startCrawl -id CrawlName [-full] [-h HostName]
[-p PortNumber] [-l true|false]
```

Where:

- `-full` (or `--full_crawl`) specifies whether to force a full crawl. If unspecified, IAS Server runs an incremental crawl. Optional
- `-id` (or `--crawl_id`) specifies the ID of the crawl to start. Required.
- `-h` (or `--host`) specifies the host where the Endeca IAS Service is running. If the flag is omitted, the default is the value of the `com.endeca.eidi.ias.server.host` property in `commandline.properties`. If the property is not set, the value then defaults to `localhost`. Optional.
- `-p` (or `--port`) specifies the port of the Endeca IAS Service. If the flag is omitted, the default is the value of the `com.endeca.eidi.ias.server.port` property in `commandline.properties`. If the property is not set, the value then defaults to `8510`. Optional.
- `-l` (or `--isPortSsl`) specifies whether to communicate with the service using an HTTPS connection. A value of `true` uses HTTPS and treats the `com.endeca.eidi.ias.server.port` property as an SSL port. A value of `false` uses HTTP and treats `com.endeca.eidi.ias.server.port` as a non-SSL port. Specify `false` if you enabled redirects from a non-SSL port to an SSL port. Optional.

To start a crawl:

1. Start a command prompt and navigate to `<install path>\IAS\<version>\bin` (for Windows), or `<install path>/IAS/<version>/bin` (for UNIX).
2. Type `ias-cmd` and specify the `startCrawl` task with the required arguments.

Note: This task name is case sensitive.

Example of starting a crawl

```
ias-cmd startCrawl -id FileCrawl
```

Stopping a crawl

The `stopCrawl` task stops a crawl.

The syntax for this task is:

```
ias-cmd stopCrawl -id CrawlName [-h HostName] [-p PortNumber] [-l true|false]
```

Where:

- `-id` (or `--crawl_id`) specifies the ID of the crawl to stop. Required.
- `-h` (or `--host`) specifies the host where the Endeca IAS Service is running. If the flag is omitted, the default is the value of the `com.endeca.eidi.ias.server.host` property in `commandline.properties`. If the property is not set, the value then defaults to `localhost`. Optional.
- `-p` (or `--port`) specifies the port of the Endeca IAS Service. If the flag is omitted, the default is the value of the `com.endeca.eidi.ias.server.port` property in `commandline.properties`. If the property is not set, the value then defaults to `8510`. Optional.
- `-l` (or `--isPortSsl`) specifies whether to communicate with the service using an HTTPS connection. A value of `true` uses HTTPS and treats the `com.endeca.eidi.ias.server.port` property as an SSL port. A value of `false` uses HTTP and treats `com.endeca.eidi.ias.server.port` as a non-SSL port. Specify `false` if you enabled redirects from a non-SSL port to an SSL port. Optional.

1. Start a command prompt and navigate to `<install path>\IAS\<version>\bin` (for Windows), or `<install path>/IAS/<version>/bin` (for UNIX).
2. Type `ias-cmd` and specify the `stopCrawl` task with the proper arguments.

Note: This task name is case sensitive.

Example of stopping a crawl

```
ias-cmd stopCrawl -id FileCrawl
```

Updating crawls

The `updateCrawls` task updates one or more existing crawl configurations with a new crawl configuration. The task does not create new crawl configurations. It updates existing crawl configurations with changes.

The syntax for this task is:

```
ias-cmd updateCrawls -f CrawlConfig.xml [-h HostName] [-p PortNumber] [-l true|false]
```

Where:

- `-f` (or `--file`) specifies the pathname of the input XML file containing the crawl configuration(s). Required.
- `-h` (or `--host`) specifies the host where the Endeca IAS Service is running. If the flag is omitted, the default is the value of the `com.endeca.eidi.ias.server.host` property in `commandline.properties`. If the property is not set, the value then defaults to `localhost`. Optional.
- `-p` (or `--port`) specifies the port of the Endeca IAS Service. If the flag is omitted, the default is the value of the `com.endeca.eidi.ias.server.port` property in `commandline.properties`, in `IAS\workspace\conf` (on Windows) or `IAS/workspace/conf` (on UNIX). If the property is not set, the value then defaults to `8510`. Optional.

- `-l` (or `--isPortSsl`) specifies whether to communicate with the service using an HTTPS connection. A value of `true` uses HTTPS and treats the `com.endeca.eidi.ias.server.port` property as an SSL port. A value of `false` uses HTTP and treats `com.endeca.eidi.ias.server.port` as a non-SSL port. Specify `false` if you enabled redirects from a non-SSL port to an SSL port. Optional.

Note: This task does not create a new crawl. The task throws an exception if you attempt to update a crawl that does not already exist.

To update crawls:

1. Start a command prompt and navigate to `<install path>\IAS\<version>\bin` (for Windows), or `<install path>/IAS/<version>/bin` (for UNIX).
2. Type `ias-cmd.bat` (for Windows), or `ias-cmd.sh` (for UNIX) and specify `updateCrawls` with the `-f` or `--file_name` flag, and the absolute path to the crawl configuration file.

   Note: This task name is case sensitive.

Example of updating crawls

```
ias-cmd updateCrawls -f C:\tmp\CrawlConfig.xml
Updated crawl FileCrawl
```

Viewing crawl status and results

The following `ias-cmd` tasks return information about crawl status and crawl results.

Getting metrics for all crawls

The `getAllCrawlMetrics` task retrieves a list of crawl IDs and their associated metrics.

The syntax for this task is:

```
ias-cmd getAllCrawlMetrics [-h HostName] [-p PortNumber] [-l true|false]
```

Where:

- `-h` (or `--host`) specifies the host where the Endeca IAS Service is running. If the flag is omitted, the default is the value of the `com.endeca.eidi.ias.server.host` property in `commandline.properties`. If the property is not set, the value then defaults to `localhost`. Optional.
- `-p` (or `--port`) specifies the port of the Endeca IAS Service. If the flag is omitted, the default is the value of the `com.endeca.eidi.ias.server.port` property in `commandline.properties`. If the property is not set, the value then defaults to `8510`. Optional.
- `-l` (or `--isPortSsl`) specifies whether to communicate with the service using an HTTPS connection. A value of `true` uses HTTPS and treats the `com.endeca.eidi.ias.server.port` property as an SSL port. A value of `false` uses HTTP and treats `com.endeca.eidi.ias.server.port` as a non-SSL port. Specify `false` if you enabled redirects from a non-SSL port to an SSL port. Optional.

To get metrics for all crawls:

1. Start a command prompt and navigate to `<install path>\IAS\<version>\bin` (for Windows), or `<install path>/IAS/<version>/bin` (for UNIX).

2. Type `ias-cmd.bat` (for Windows), or `ias-cmd.sh` (for UNIX) and specify the `getAllCrawlMetrics` task.

Note: This task name is case sensitive.

Example of getting metrics for all crawls

```
ias-cmd getAllCrawlMetrics
Metrics for crawl FileCrawl:

ARCHIVED_DIRECTORIES_CRAWLED:   0
ARCHIVED_DIRECTORIES_FILTERED:  0
ARCHIVED_FILES_CRAWLED: 0
ARCHIVED_FILES_FILTERED:        0
CRAWL_MODE:     FULL_CRAWL
CRAWL_STATE:    NOT_RUNNING
CRAWL_STOP_CAUSE:    COMPLETED
DELETED_RECORDS:        0
DIRECTORIES_CRAWLED:    3009
DIRECTORIES_FILTERED:   0
DURATION_IN_SECONDS:    595
END_TIME:       Thu Apr 23 13:46:27 EDT 2009
FAILED_TEXT_EXTRACTIONS:        65
FILES_CRAWLED:  28849
FILES_FILTERED: 0
NEW_OR_UPDATED_RECORDS: 31858
NONARCHIVED_DIRECTORIES_CRAWLED:        3009
NONARCHIVED_DIRECTORIES_FILTERED:       0
NONARCHIVED_FILES_CRAWLED:      28849
NONARCHIVED_FILES_FILTERED:     0
START_TIME:     Thu Apr 23 13:36:32 EDT 2009
SUCCESSFUL_TEXT_EXTRACTIONS:    1420
SUCCESSFUL_TEXT_EXTRACTIONS_AFTER_RETRY:        1
TOTAL_RECORDS:  31858

Metrics for crawl FileCrawl2:

ARCHIVED_DIRECTORIES_CRAWLED:   3787
ARCHIVED_DIRECTORIES_FILTERED:  0
ARCHIVED_FILES_CRAWLED: 62085
ARCHIVED_FILES_FILTERED:        0
CRAWL_MODE:     FULL_CRAWL
CRAWL_STATE:    NOT_RUNNING
CRAWL_STOP_CAUSE:    COMPLETED
DELETED_RECORDS:        0
DIRECTORIES_CRAWLED:    16504
DIRECTORIES_FILTERED:   0
DURATION_IN_SECONDS:    1569
END_TIME:       Thu Apr 23 14:37:53 EDT 2009
FAILED_TEXT_EXTRACTIONS:        67
FILES_CRAWLED:  153511
FILES_FILTERED: 0
NEW_OR_UPDATED_RECORDS: 170015
NONARCHIVED_DIRECTORIES_CRAWLED:        12717
NONARCHIVED_DIRECTORIES_FILTERED:       0
NONARCHIVED_FILES_CRAWLED:      91426
NONARCHIVED_FILES_FILTERED:     0
START_TIME:     Thu Apr 23 14:11:44 EDT 2009
SUCCESSFUL_TEXT_EXTRACTIONS:    7109
SUCCESSFUL_TEXT_EXTRACTIONS_AFTER_RETRY:        1
TOTAL_RECORDS:  170015
```

Getting the metrics for a crawl

The `getCrawlMetrics` task retrieves metrics for a particular crawl.

The syntax for this task is:

```
ias-cmd getCrawlMetrics -id CrawlName [-h HostName] [-p PortNumber] [-l true|false]
```

Where:

- `-id` (or `--crawl_id`) specifies the name of the crawl for to retrieve metrics for. Required.
- `-h` (or `--host`) specifies the host where the Endeca IAS Service is running. If the flag is omitted, the default is the value of the `com.endeca.eidi.ias.server.host` property in `commandline.properties`. If the property is not set, the value then defaults to `localhost`. Optional.
- `-p` (or `--port`) specifies the port of the Endeca IAS Service. If the flag is omitted, the default is the value of the `com.endeca.eidi.ias.server.port` property in `commandline.properties`. If the property is not set, the value then defaults to `8510`. Optional.
- `-l` (or `--isPortSsl`) specifies whether to communicate with the service using an HTTPS connection. A value of `true` uses HTTPS and treats the `com.endeca.eidi.ias.server.port` property as an SSL port. A value of `false` uses HTTP and treats `com.endeca.eidi.ias.server.port` as a non-SSL port. Specify `false` if you enabled redirects from a non-SSL port to an SSL port. Optional.

1. Start a command prompt and navigate to `<install path>\IAS\<version>\bin` (for Windows), or `<install path>/IAS/<version>/bin` (for UNIX).
2. Type `ias-cmd.bat` (for Windows), or `ias-cmd.sh` (for UNIX) and specify the `getCrawlMetrics` task with the id of the crawl for which you want to get metrics.

Note: This task name is case sensitive.

Example of getting the metrics for a crawl

```
ias-cmd getCrawlMetrics -id Test
ARCHIVED_DIRECTORIES_CRAWLED:   0
ARCHIVED_DIRECTORIES_FILTERED:  0
ARCHIVED_FILES_CRAWLED: 0
ARCHIVED_FILES_FILTERED:        0
CRAWL_MODE:     FULL_CRAWL
CRAWL_STATE:    NOT_RUNNING
CRAWL_STOP_CAUSE:       COMPLETED
DELETED_RECORDS:        0
DIRECTORIES_CRAWLED:    97
DIRECTORIES_FILTERED:   0
DURATION_IN_SECONDS:    25
END_TIME:       Thu Jan 07 16:33:17 EST 2010
FAILED_RECORDS: 0
FAILED_TEXT_EXTRACTIONS:        0
FILES_CRAWLED:  688
FILES_FILTERED: 0
NEW_OR_UPDATED_RECORDS: 785
NONARCHIVED_DIRECTORIES_CRAWLED:        97
NONARCHIVED_DIRECTORIES_FILTERED:       0
NONARCHIVED_FILES_CRAWLED:      688
NONARCHIVED_FILES_FILTERED:     0
START_TIME:     Thu Jan 07 16:32:51 EST 2010
SUCCESSFUL_TEXT_EXTRACTIONS:    557
SUCCESSFUL_TEXT_EXTRACTIONS_AFTER_RETRY:        0
TOTAL_RECORDS:  785
```

Getting the status of a crawl

The `getCrawlStatus` task returns the status of a specific crawl.

The syntax for this task is:

```
ias-cmd getCrawlStatus -id CrawlName [-h HostName] [-p PortNumber] [-l true|false]
```

Where:

- `-id` (or `--crawl_id`) specifies the name of the crawl to retrieve status for. Required.
- `-h` (or `--host`) specifies the host where the Endeca IAS Service is running. If the flag is omitted, the default is the value of the `com.endeca.eidi.ias.server.host` property in `commandline.properties`. If the property is not set, the value then defaults to `localhost`. Optional.
- `-p` (or `--port`) specifies the port of the Endeca IAS Service. If the flag is omitted, the default is the value of the `com.endeca.eidi.ias.server.port` property in `commandline.properties`. If the property is not set, the value then defaults to `8510`. Optional.
- `-l` (or `--isPortSsl`) specifies whether to communicate with the service using an HTTPS connection. A value of `true` uses HTTPS and treats the `com.endeca.eidi.ias.server.port` property as an SSL port. A value of `false` uses HTTP and treats `com.endeca.eidi.ias.server.port` as a non-SSL port. Specify `false` if you enabled redirects from a non-SSL port to an SSL port. Optional.

To get the status of a crawl:

1. Start a command prompt and navigate to `<install path>\IAS\<version>\bin` (for Windows), or `<install path>/IAS/<version>/bin` (for UNIX).
2. Type `ias-cmd.bat` (for Windows), or `ias-cmd.sh` (for UNIX) and specify the `getCrawlStatus` task with the id of the acquisition.

   Note: This task name is case sensitive.

Example of getting the status of a crawl

```
ias-cmd getCrawlStatus -id FileCrawl
RUNNING
```

Chapter 7

Component Instance Manager Command-line Utility

This section describes how to run the tasks of the Component Instance Manager (CIM) Command-line Utility.

Overview of the CIM Command-line Utility

The Component Instance Manager (CIM) Command-line Utility is a tool to create components, delete components, and view components. The Endeca IAS Service must be running before you can execute any of the CIM Command-line Utility tasks.

In this version of IAS, the types of components you can manage with the CIM Command-line Utility are Record Store components. In future releases, the utility may be extended to manage additional types of components.

The CIM Command-line Utility is a script named `component-manager-cmd.sh` (for Linux/UNIX systems) and `component-manager-cmd.bat` (on Windows) that you run from a command prompt. The script is in the `bin` directory.

Help options

The CIM Command-line Utility has two help options that display the usage syntax. The `--help` option displays a summary of the tasks. The `--help-detail` option displays detailed usage information for all the tasks.

For example:

```
C:Oracle\Endeca\IAS\3.0.0\bin>component-manager-cmd.bat --help
usage: component-manager-cmd <task-name> [options]

  list-types
  list-components
  create-component
  delete-component

For detailed usage information including task options, use --help-detail
For detailed usage information for individual task options, use <task-name> --help
```

Command-line options

The command syntax for executing the tasks is:

```
component-manager-cmd task-name [options]
```

The *task-name* argument is the task to be performed by the utility, such as the `createCrawls` task. The task options vary, depending on the task. However, these options can be used with any task:

- `-h` (or `--host`) specifies the host name of the machine on which the IAS Service is running. If the flag is omitted, it defaults to the value of the `com.endeca.eidi.ias.server.host` property in `workspace\IAS\conf\commandline.properties`. If the property is not set, the value then defaults to `localhost` as the host name.
- `-p` (or `--port`) specifies the port on which IAS Service is listening. If the flag is omitted, it defaults to the value of the `com.endeca.eidi.ias.server.port` property in `<install path>\workspace\IAS\conf\commandline.properties`. If the property is not set, the value then defaults to `8510` as the port number.
- `-l` (or `--isPortSsl`) specifies whether to communicate with the service using an HTTPS connection. A value of `true` uses HTTPS and treats the `com.endeca.eidi.ias.server.port` property as an SSL port. A value of `false` uses HTTP and treats `com.endeca.eidi.ias.server.port` as a non-SSL port. Specify `false` if you enabled redirects from a non-SSL port to an SSL port. Optional.

Host and port settings

You first specify the host and port settings for the Endeca IAS Service as part of the installation wizard. That host runs the IAS Server, the Component Instance Manager, and all Record Store instances. The installation wizard then writes the host (`com.endeca.eidi.ias.server.host`) and port (`com.endeca.eidi.ias.server.port`) settings as properties in `commandline.properties`. All of the command-line utilities use these settings as default values if you omit the `-h` and `-p` flags when executing any tasks.

Setting the bin directory in the PATH environment variable

Although not required, it is recommended that you set the path of the `bin` directory in your systems' PATH environment variable. This allows you to run the Component Instance Manager Command-line Utility script from any location.

About error handling

- If desired, you can re-configure the default logging settings in `<install path>\IAS\workspace\conf\component-manager-cmd.log4j.properties`.
- Errors print to standard error, unless you redirect `std err` to a file instead.
- Errors of mis-configured command-line tasks or incorrect input parameters are written to standard out.

Creating a Record Store

The `create-component` task creates a Record Store instance.

The syntax for this task is:

```
component-manager-cmd create-component -n RecordStoreName  -t RecordStore
```

```
[-h HostName] [-p PortNumber] [-l true|false]
```

Where:

- `-n` specifies the name of the component you are creating. Required.
- `-t` specifies the type of the component instance you want to create. Specify `RecordStore`. Required.
- `-h` (or `--host`) specifies the host where the IAS Service is running. If the flag is omitted, the default is the value of the `com.endeca.eidi.ias.server.host` property in `commandline.properties`. If the property is not set, the value then defaults to `localhost`. Optional.
- `-p` (or `--port`) specifies the port of the IAS Service. If the flag is omitted, the default is the value of the `com.endeca.eidi.ias.server.port` property in `commandline.properties`. If the property is not set, the value then defaults to `8510`. Optional.
- `-l` (or `--isPortSsl`) specifies whether to communicate with the service using an HTTPS connection. A value of `true` uses HTTPS and treats the `com.endeca.eidi.ias.server.port` property as an SSL port. A value of `false` uses HTTP and treats `com.endeca.eidi.ias.server.port` as a non-SSL port. Specify `false` if you enabled redirects from a non-SSL port to an SSL port. Optional.

To create a Record Store:

1. Start a command prompt and navigate to `<install path>\IAS\<version>\bin`.
2. Run the `create-component` task and specify the `-t` option with an argument of `RecordStore` and specify the`-n` option with a Record Store instance name of your choice.

Example of creating a Record Store

This example creates a Record Store named RS1:

```
component-manager-cmd.bat create-component -n RS1 -t RecordStore
```

Deleting a Record Store

The `delete-component` task deletes a Record Store.

The syntax for this task is:

```
component-manager-cmd delete-component -n RecordStoreName
[-h HostName] [-p PortNumber] [-l true|false]
```

where:

- `-n` specifies the name of the component you are deleting. Required.
- `-h` (or `--host`) specifies the host where the Component Instance Manager is running. If the flag is omitted, the default is the value of the `com.endeca.eidi.ias.server.host` property in `commandline.properties`. If the property is not set, the value then defaults to `localhost`. Optional.
- `-p` (or `--port`) specifies the port of the Component Instance Manager. If the flag is omitted, the default is the value of the `com.endeca.eidi.ias.server.port` property in `commandline.properties`. If the property is not set, the value then defaults to `8510`. Optional.
- `-l` (or `--isPortSsl`) specifies whether to communicate with the service using an HTTPS connection. A value of `true` uses HTTPS and treats the `com.endeca.eidi.ias.server.port` property as an SSL port. A value of `false` uses HTTP and treats `com.endeca.eidi.ias.server.port` as a non-SSL port. Specify `false` if you enabled redirects from a non-SSL port to an SSL port. Optional.

To delete a Record Store:

1. Start a command prompt and navigate to <*install path*>\IAS\<version>\bin.
2. Run the `delete-component` task and specify the `-n` option.

Example of deleting a Record Store

This example deletes a Record Store named RS1:

```
component-manager-cmd.bat
delete-component -n RS1
```

Listing components

The `list-components` task lists all component instances that are managed by the Component Instance Manager. Executing the task returns a list of all managed components in the IAS Service.

The syntax for this task is:

```
component-manager-cmd list-components [-h HostName] [-p PortNumber]
[-l true|false]
```

where:

- `-h` (or `--host`) specifies the host where the Component Instance Manager is running. If the flag is omitted, the default is the value of the `com.endeca.eidi.ias.server.host` property in `commandline.properties`. If the property is not set, the value then defaults to `localhost`. Optional.
- `-p` (or `--port`) specifies the port of the Component Instance Manager. If the flag is omitted, the default is the value of the `com.endeca.eidi.ias.server.port` property in `commandline.properties`. If the property is not set, the value then defaults to `8510`. Optional.
- `-l` (or `--isPortSsl`) specifies whether to communicate with the service using an HTTPS connection. A value of `true` uses HTTPS and treats the `com.endeca.eidi.ias.server.port` property as an SSL port. A value of `false` uses HTTP and treats `com.endeca.eidi.ias.server.port` as a non-SSL port. Specify `false` if you enabled redirects from a non-SSL port to an SSL port. Optional.

To list components:

1. Start a command prompt and navigate to <*install path*>\IAS\<version>\bin.
2. Run the `list-components` task with any of the options listed above.

Example of listing components

This example lists the Record Store instances for a sample application running in the Endeca IAS Service:

```
component-manager-cmd.bat list-components
NAME            TYPE            STATUS
ebizsampleapp-trigger-dimensions        RecordStore     RUNNING
ebizsampleapp-products  RecordStore     RUNNING
ebizsampleapp-category-dimension        RecordStore     RUNNING
```

If no components have been created, the `list-components` task returns the following:

```
No components have been provisioned.
```

Listing types

The `list-types` task lists all component types that are managed by the Component Instance Manager. Executing the task returns a list of all managed component types in the IAS Service.

In this release, the only supported component type is `RecordStore`.

The syntax for this task is:

```
component-manager-cmd list-types [-h HostName] [-p PortNumber] [-l true|false]
```

where:

- `-h` (or `--host`) specifies the host where the Component Instance Manager is running. If the flag is omitted, the default is the value of the `com.endeca.eidi.ias.server.host` property in `commandline.properties`. If the property is not set, the value then defaults to `localhost`. Optional.
- `-p` (or `--port`) specifies the port of the Component Instance Manager. If the flag is omitted, the default is the value of the `com.endeca.eidi.ias.server.port` property in `commandline.properties`. If the property is not set, the value then defaults to `8510`. Optional.
- `-l` (or `--isPortSsl`) specifies whether to communicate with the service using an HTTPS connection. A value of `true` uses HTTPS and treats the `com.endeca.eidi.ias.server.port` property as an SSL port. A value of `false` uses HTTP and treats `com.endeca.eidi.ias.server.port` as a non-SSL port. Specify `false` if you enabled redirects from a non-SSL port to an SSL port. Optional.

To list component types:

1. Start a command prompt and navigate to `<install path>\IAS\<version>\bin`.
2. Run the `list-types` task with any of the options listed above.

Example of listing types

This example lists the type of components running on the Endeca IAS Service:

```
component-manager-cmd.bat list-types
ID              PATH
RecordStore     recordstore-core-3.0.0.jar
```

Chapter 8

Record Store Command-line Utility

This section describes how to run the tasks of the Record Store Command-line Utility.

Overview of the Record Store Command-line Utility

The Record Store Command-line Utility provides the ability to read records from and write records to a Record Store instance, in addition to a number of utility tasks such as setting client IDs and rolling back transactions.

The Record Store Command-line Utility is a script named `recordstore-cmd.sh` (for Linux/UNIX systems) and `recordstore-cmd.bat` (for Windows) that you run from a command prompt. The scripts are in the `bin` directory.

Transactions

Read and write operations take place within the scope of a transaction. You can specify the start, commit, or roll back of a transaction. This is useful in cases where you want to perform multiple operations within the scope of a single transaction. If you do not explicitly control the transaction, all read and write operations take place in a default auto commit mode.

Help options

The Record Store Command-line Utility has two help options that display the usage syntax. The `--help` option displays a summary of the tasks. The `--help-detail` option displays detailed usage information for all the tasks. For example:

```
C:\Oracle\Endeca\IAS\3.0.0\bin>recordstore-cmd --help
usage: recordstore-cmd <task-name> [options]

[READ TASKS]
  read-baseline
  read-delta
  read-by-id
[UTILITY TASKS]
  clean
  clear-last-read-generation
  commit-transaction
  get-configuration
  get-last-committed-generation
  get-last-read-generation
  get-write-generation
  list-active-transactions
```

```
  list-client-states
  list-generations
  rollback-transaction
  set-configuration
  set-last-read-generation
  start-transaction
[WRITE TASKS]
  write

For detailed usage information including task options, use --help-detail
For detailed usage information for individual task options, use <task-name> --help
```

Command-line options

With one exception, the command syntax for executing the tasks is:

```
recordstore-cmd task-name [options]
```

The exception to this syntax format is the `read-by-id` task, which is explained in its own topic.

The *task-name* argument is the task to be performed by the utility, such as the `read-delta` task. The task options vary, depending on the task. However, these options can be used with any task:

- `-h` (or `--host`) specifies the host name of the machine on which the Record Store is running. If the flag is omitted, it defaults to the value of the `com.endeca.eidi.ias.server.host` property in `commandline.properties`. If the property is not set, the value then defaults to `localhost` as the host name.
- `-p` (or `--port`) specifies the port on which the Record Store is listening. If the flag is omitted, it defaults to the value of the `com.endeca.eidi.ias.server.port` property in `commandline.properties`. If the property is not set, the value then defaults to `8510` as the port number.
- `-l` (or `--isPortSsl`) specifies whether to communicate with the service using an HTTPS connection. A value of `true` uses HTTPS and treats the `com.endeca.eidi.ias.server.port` property as an SSL port. A value of `false` uses HTTP and treats `com.endeca.eidi.ias.server.port` as a non-SSL port. Specify `false` if you enabled redirects from a non-SSL port to an SSL port. Optional.

Host and port settings

You first specify the host and port settings for the Endeca IAS Service as part of the installation wizard. That host runs the IAS Server, the Component Instance Manager, and all Record Store instances. The installation wizard then writes the host (`com.endeca.eidi.ias.server.host`) and port (`com.endeca.eidi.ias.server.port`) settings as properties in `commandline.properties`. All of the command-line utilities use these settings as default values if you omit the `-h` and `-p` flags when executing any tasks.

Setting the bin directory in the PATH environment variable

Although not required, it is recommended that you set the path of the `bin` directory in your systems' PATH environment variable. This allows you to run the Record Store Command-line Utility script from any location.

About error handling

- If desired, you can re-configure the default logging settings in `<install path>\IAS\workspace\conf\recordstore-cmd.log4j.properties`.
- By default, errors print to a log file named `recordstore-cmd.log` that is located in the `logs` directory.

Writing tasks

The following `recordstore-cmd` tasks perform write operations to a Record Store instance.

Writing records

The `write` task writes a list of records into a specified Record Store instance.

The syntax for this task is:

```
recordstore-cmd write -a RecordStoreInstanceName [-b]
-f InputFile [-h HostName] [-l true|false] [-p PortNumber] [-r Type] [-x Id]
```

where:

- `-a` (or `--instanceName`) specifies the name of a Record Store instance. Required.
- `-b` (or `--baseline`) is a flag with no arguments that specifies that this is to be a baseline write. If the Record Store has any existing generations, a baseline write will not delete those previous generations, however, it will mark them as "to be deleted" and the cleaner will delete them when it runs (if the records are older than the generation retention time). If the flag is omitted, the write operation is considered an incremental write to the last-committed generation. Optional.
- `-f` (or `--file`) specifies the file that contains Endeca records. The filename extension will determine the format of the input file. Valid extensions for the file are `.xml` (for an XML format) and `.bin` (for a binary format); either file type can also have an additional, optional `.gz` extension if it is a compressed file. Required.
- `-h` (or `--host`) specifies the host where the Endeca IAS Service is running. If the flag is omitted, the default is the value of the `com.endeca.eidi.ias.server.host` property in `commandline.properties`. If the property is not set, the value then defaults to `localhost`. Optional.
- `-l` (or `--isPortSsl`) specifies whether to communicate with the service using an HTTPS connection. A value of `true` uses HTTPS and treats the `com.endeca.eidi.ias.server.port` property as an SSL port. A value of `false` uses HTTP and treats `com.endeca.eidi.ias.server.port` as a non-SSL port. Specify `false` if you enabled redirects from a non-SSL port to an SSL port. Optional.
- `-p` (or `--port`) specifies the port of the Endeca IAS Service. If the flag is omitted, the default is the value of the `com.endeca.eidi.ias.server.port` property in `commandline.properties`. If the property is not set, the value then defaults to `8510`. Optional.
- `-x` (or `--transaction`) specifies the active transaction ID to use. If you use this option, you must follow the command with a `commit-transaction` task to commit the write operation. If this flag is omitted, the operation is done in auto-commit mode. Optional.

Examples of writing records

If there are two generations in the Record Store, this command:

```
recordstore-cmd write -a RS2 -b -f basedata.xml
```

will write the records in the `basedata.xml` file as a baseline write operation. If you check the log output, you should see messages similar to these:

```
Starting new transaction with generation Id 3
Started transaction 10 of type READ_WRITE
Processing delete all for generation 3
Marking generation committed: 3
```

```
Committed transaction 10
```

The Delete message (`Processing delete all for generation 3`) indicates that the transaction that created Generation 3 also marked the previous generations for deletion.

If you then perform a subsequent incremental write command:

```
recordstore-cmd write -f incrdata.xml
```

the console or log output messages should look like these:

```
Starting new transaction with generation Id 4
Started transaction 11 of type READ_WRITE
Marking generation committed: 4
Committed transaction 11
```

At this point, the Record Store has two generations: Generation 3 is a baseline generation and Generation 4 is an incremental generation. If you then run a baseline update, it will use both generations.

Reading tasks

The following `recordstore-cmd` tasks perform read operations from a Record Store instance.

Reading baselines

The `read-baseline` task reads the baseline records from a Record Store instance.

The syntax for this task is:

```
recordstore-cmd read-baseline -a RecordStoreInstanceName
[-c] [-f FileName.xml] [-g GenId] [-h HostName] [-l true|false]
[-p PortNumber] [-n NumRecs] [-x id]
```

where:

- `-a` (or `--instanceName`) specifies the name of a Record Store instance. Required.
- `-c` (or `--count`) that only prints the record count from the read. Optional.
- `-f` (or `--file`) specifies the pathname of the file to which the Endeca records will be output. The filename extension determines the format of the output file. Valid extensions for the file are `.xml` (for an XML format) and `.bin` (for a binary format); the file can also have an additional, optional `.gz` extension if it is a compressed file. If unspecified, the record are written to the console. Optional.
- `-g` (or `--generation`) specifies the ID of the generation from which the records are read. If omitted, records from the last-committed generation are read. Optional.
- `-h` (or `--host`) specifies the host where the Endeca IAS Service is running. If the flag is omitted, the default is the value of the `com.endeca.eidi.ias.server.host` property in `commandline.properties`. If the property is not set, the value then defaults to `localhost`. Optional.
- `-l` (or `--isPortSsl`) specifies whether to communicate with the service using an HTTPS connection. A value of `true` uses HTTPS and treats the `com.endeca.eidi.ias.server.port` property as an SSL port. A value of `false` uses HTTP and treats `com.endeca.eidi.ias.server.port` as a non-SSL port. Specify `false` if you enabled redirects from a non-SSL port to an SSL port. Optional.
- `-p` (or `--port`) specifies the port of the Endeca IAS Service. If the flag is omitted, the default is the value of the `com.endeca.eidi.ias.server.port` property in `commandline.properties`. If the property is not set, the value then defaults to `8510`. Optional.

- `-n` (or `--firstN`) specifies that only the first *numRecs* records of the baseline will be read. If omitted, all records are read. Optional.
- `-x` (or `--transaction`) specifies the active transaction ID to use. If you use this option, you must follow it with a `commit-transaction` task to commit the read operation. If this flag is omitted, the operation is done in auto-commit mode. Optional.

Examples of reading baselines

The first example reads the first 50 baseline records (from the last-committed generation) and outputs them to a file:

```
recordstore-cmd read-baseline -a RS1 -n 50 -f c:\recdata\basedata.xml
```

The output is written in an XML format to the `basedata.xml` file located in the `C:\recdata` directory.

The second example prints the number of records in the baseline:

```
recordstore-cmd read-baseline -a RS1 -c -g 2
```

The command prints out the number of records in generation 2 of the Record Store.

Reading delta records

The `read-delta` task reads the delta between two or more generations in the Record Store.

Delta records can be one of three types:

- Modified records. A modified record has the same record ID as the previous version, but the content (as determined from the `changePropertyNames` property) has changed.
- Added records. An added (new) record will have a record ID that does not appear in the previous generations.
- Deleted records. A deleted record will have a valid record ID, but its `Endeca.Action` property will be set to `DELETE`.

The syntax for this task is:

```
recordstore-cmd read-delta -a RecordStoreInstanceName [-c] [-f FileName]
[-n NumRecs] [-h HostName] [-l true|false] [-p PortNumber] [-s StartGenId]
[-e EndGenId] [-x Id]
```

where:

- `-a` (or `--instanceName`) specifies the name of a Record Store instance. Required.
- `-c` (or `--count`) prints the record count from the read. Optional.
- `-f` (or `--file`) specifies the pathname of the file to which the Endeca records will be output. The filename extension will determine the format of the output file. Valid extensions for the file are `.xml` (for an XML format) and `.bin` (for a binary format); the file can also have an additional, optional `.gz` extension if it is a compressed file. Optional.
- `-n` (or `--firstN`) specifies that only the first *numRecs* number of delta records will be read. If omitted, all records are read. Optional.
- `-h` (or `--host`) specifies the host where the Endeca IAS Service is running. If the flag is omitted, the default is the value of the `com.endeca.eidi.ias.server.host` property in `commandline.properties`. If the property is not set, the value then defaults to `localhost`. Optional.

- `-l` (or `--isPortSsl`) specifies whether to communicate with the service using an HTTPS connection. A value of `true` uses HTTPS and treats the `com.endeca.eidi.ias.server.port` property as an SSL port. A value of `false` uses HTTP and treats `com.endeca.eidi.ias.server.port` as a non-SSL port. Specify `false` if you enabled redirects from a non-SSL port to an SSL port. Optional.
- `-p` (or `--port`) specifies the port of the Endeca IAS Service. If the flag is omitted, the default is the value of the `com.endeca.eidi.ias.server.port` property in `commandline.properties`. If the property is not set, the value then defaults to `8510`. Optional.
- `-s` (or `--startGeneration`) specifies the ID of the start generation from which the diff will be done. If omitted, the initial generation is used. Optional.
- `-e` (or `--endGeneration`) specifies the ID of the end generation from which the diff will be done. If omitted, the last-committed generation is used. Optional.
- `-x` (or `--transaction`) specifies the active transaction ID to use. If you use this option, you must follow it with a `commit-transaction` task to commit the read operation. If this flag is omitted, the operation is done in auto-commit mode. Optional.

For this task, it is recommended that you explicitly specify the start and end generations.

Example of reading delta records

This example reads all the delta records that constitute the difference between Generation 1 and Generation 2 and writes them to a file:

```
recordstore-cmd read-delta -a RS1 -f c:\recdata\diffdata.xml -s 1 -e 2
```

The delta records are written in an XML format to the `diffdata.xml` file located in the `C:\recdata` directory.

If you only want a record count of the difference, use the `-c` option:

```
recordstore-cmd read-delta -a RS1 -c -s 1 -e 2
```

The number of delta records read is output to the console.

Reading specific records

The `read-by-id` task reads one or more specific records from a Record Store instance.

The `read-by-id` task uses a syntax that is different from the other tasks. The difference is that you specify the record IDs at the end of the command line (i.e., after all the options have been specified).

The syntax for this task is:

```
recordstore-cmd read-by-id -a RecordStoreInstanceName [-c] [-f FileName]
[-g GenId] [-h HostName] [-l true|false] [-p PortNumber] [-x Id]
[RecId1 [RecId2 [... RecIdN]
```

where:

- `-a` (or `--instanceName`) specifies the name of a Record Store instance. Required.
- `-c` (or `--count`) prints the record count from the read. Optional.
- `-f` (or `--file`) specifies the pathname of the file to which the Endeca records are output. The filename extension determines the format of the output file. Valid extensions for the file are `.xml` (for an XML format) and `.bin` (for a binary format); the file can also have an additional, optional `.gz` extension to create a compressed file. Optional.

- `-g` (or `--generation`) specifies the ID of the generation from which the records are read. If omitted, records from the last-committed generation are read. Optional.
- `-h` (or `--host`) specifies the host where the Endeca IAS Service is running. If the flag is omitted, the default is the value of the `com.endeca.eidi.ias.server.host` property in `commandline.properties`. If the property is not set, the value then defaults to `localhost`. Optional.
- `-l` (or `--isPortSsl`) specifies whether to communicate with the service using an HTTPS connection. A value of `true` uses HTTPS and treats the `com.endeca.eidi.ias.server.port` property as an SSL port. A value of `false` uses HTTP and treats `com.endeca.eidi.ias.server.port` as a non-SSL port. Specify `false` if you enabled redirects from a non-SSL port to an SSL port. Optional.
- `-p` (or `--port`) specifies the port of the Endeca IAS Service. If the flag is omitted, the default is the value of the `com.endeca.eidi.ias.server.port` property in `commandline.properties`. If the property is not set, the value then defaults to `8510`. Optional.
- `-x` (or `--transaction`) specifies the active transaction ID to use. If you use this option, you must follow it with a `commit-transaction` task to commit the read operation. If this flag is omitted, the operation is done in auto-commit mode. Optional.
- `recId` is the ID of the record to read. The record ID is the value of the record property being used for the `idPropertyName` configuration property. For multiple records, you must specify a space-delimited list of record IDs. If an ID contains a space, enclose the ID within double quotation marks.

Example of reading a specific record

Assume that the `idPropertyName` configuration in the Record Store instance is set to the `Endeca.Web.URL` record property. Also assume that you want to read a record that has this value:

```
<PROP NAME="Endeca.Web.URL">
   <PVAL>http://endeca.com/contact.html</PVAL>
</PROP>
```

This means that the string `http://endeca.com/contact.html` is the ID of that record. You would therefore retrieve that record with this command:

```
recordstore-cmd read-by-id -a RS1 -f rec.xml http://endeca.com/contact.html
```

The record will be written in an XML format to the `rec.xml` file.

Utility tasks

The following `recordstore-cmd` tasks perform utility operations to manage a Record Store instance.

Cleaning a Record Store instance

The `clean` task manually removes stale generations of records from a specified Record Store instance.

By default, the `clean` task runs automatically as a background process, at time intervals specified by the `cleanerInterval` configuration property. The `clean` task automatically removes record generations that exceed the `generationRetentionTime` configuration property.

The task syntax is:

```
recordstore-cmd clean -a RecordStoreInstanceName [-h HostName]
[-l true|false] [-p PortNumber]
```

where:

- `-a` (or `--instanceName`) specifies the name of a Record Store instance. Required.
- `-h` (or `--host`) specifies the host where the Endeca IAS Service is running. If the flag is omitted, the default is the value of the `com.endeca.eidi.ias.server.host` property in `commandline.properties`. If the property is not set, the value then defaults to `localhost`. Optional.
- `-l` (or `--isPortSsl`) specifies whether to communicate with the service using an HTTPS connection. A value of `true` uses HTTPS and treats the `com.endeca.eidi.ias.server.port` property as an SSL port. A value of `false` uses HTTP and treats `com.endeca.eidi.ias.server.port` as a non-SSL port. Specify `false` if you enabled redirects from a non-SSL port to an SSL port. Optional.
- `-p` (or `--port`) specifies the port of the Endeca IAS Service. If the flag is omitted, the default is the value of the `com.endeca.eidi.ias.server.port` property in `commandline.properties`. If the property is not set, the value then defaults to `8510`. Optional.

Note that only one `clean` process can run at a time. If a `clean` task is running when you issue this command, an exception is thrown and the second `clean` process does not run.

There are several cases where the `clean` task does not remove eligible generations in a Record Store instance:

- If it is the only generation in a Record Store instance.
- If the generation is in use.
- If it is the last committed generation.
- If it is the last generation read by a client, such as a Record Store Reader component (Endeca Integrator) or the Record Store API.

Clearing the last read generation

The `clear-last-read-generation` task clears the last-read generation for a given client ID. This task is the counterpart of `set-last-read-generation`.

The syntax for this task is:

```
recordstore-cmd clear-last-read-generation  -a RecordStoreInstanceName
-c ClientId [-h HostName] [-l true|false] [-p PortNumber] [-x Id]
```

where:

- `-a` (or `--instanceName`) specifies the name of a Record Store instance. Required.
- `-c` (or `--client`) specifies a string to identify the client ID. Required.
- `-h` (or `--host`) specifies the host where the Endeca IAS Service is running. If the flag is omitted, the default is the value of the `com.endeca.eidi.ias.server.host` property in `commandline.properties`. If the property is not set, the value then defaults to `localhost`. Optional.
- `-l` (or `--isPortSsl`) specifies whether to communicate with the service using an HTTPS connection. A value of `true` uses HTTPS and treats the `com.endeca.eidi.ias.server.port` property as an SSL port. A value of `false` uses HTTP and treats `com.endeca.eidi.ias.server.port` as a non-SSL port. Specify `false` if you enabled redirects from a non-SSL port to an SSL port. Optional.
- `-p` (or `--port`) specifies the port of the Endeca IAS Service. If the flag is omitted, the default is the value of the `com.endeca.eidi.ias.server.port` property in `commandline.properties`. If the property is not set, the value then defaults to `8510`. Optional.

- `-x` (or `--transaction`) specifies the active transaction ID to use. If you use this option, you must follow it with a `commit-transaction` task to commit the read operation. If this flag is omitted, the operation is done in auto-commit mode. Optional.

Example of setting the last-read generation

This example clears the last read generation flag for the client ID `rsreader1`:

```
recordstore-cmd clear-last-read-generation -a RS1 -c rsreader1
```

Committing transactions

The `commit-transaction` task commits an active (uncommitted) transaction for a specified Record Store instance.

The syntax for this task is:

```
recordstore-cmd commit-transaction -a RecordStoreInstanceName -x Id
[-h HostName] [-l true|false] [-p PortNumber]
```

where:

- `-a` (or `--instanceName`) specifies the name of a Record Store instance. Required.
- `-h` (or `--host`) specifies the host where the Endeca IAS Service is running. If the flag is omitted, the default is the value of the `com.endeca.eidi.ias.server.host` property in `commandline.properties`. If the property is not set, the value then defaults to `localhost`. Optional.
- `-l` (or `--isPortSsl`) specifies whether to communicate with the service using an HTTPS connection. A value of `true` uses HTTPS and treats the `com.endeca.eidi.ias.server.port` property as an SSL port. A value of `false` uses HTTP and treats `com.endeca.eidi.ias.server.port` as a non-SSL port. Specify `false` if you enabled redirects from a non-SSL port to an SSL port. Optional.
- `-p` (or `--port`) specifies the port of the Endeca IAS Service. If the flag is omitted, the default is the value of the `com.endeca.eidi.ias.server.port` property in `commandline.properties`. If the property is not set, the value then defaults to `8510`. Optional.
- `-x` (or `--transaction`) specifies the ID of the transaction that will be committed. Required.

Example of committing a transaction

This example commits the transaction with an ID of 8:

```
recordstore-cmd commit-transaction -a RS1 -x 8
```

If the command succeeds, it prints the following message:

```
Committed transaction: 8
```

If the command fails, it prints the following error message:

```
Failed to commit transaction: 8
```

Getting the configuration of a Record Store instance

The `get-configuration` task returns the configuration settings of a specified Record Store instance.

A Record Store instance has a default configuration that you can retrieve and save. You can modify the configuration and use it to configure a new Record Store or reconfigure an existing Record Store instance.

The syntax for this task is:

```
recordstore-cmd get-configuration -a RecordStoreInstanceName
[-f FileName.xml] [-h HostName] [-l true|false] [-n] [-p PortNumber]
```

where:

- `-a` (or `--instanceName`) specifies the name of a Record Store instance. Required.
- `-f` (or `--file`) specifies the XML file name where you want to save the configuration settings. Omitting this option sends the XML for the configuration settings to stdout. Optional.
- `-h` (or `--host`) specifies the host where the Endeca IAS Service is running. If the flag is omitted, the default is the value of the `com.endeca.eidi.ias.server.host` property in `commandline.properties`. If the property is not set, the value then defaults to `localhost`. Optional.
- `-l` (or `--isPortSsl`) specifies whether to communicate with the service using an HTTPS connection. A value of `true` uses HTTPS and treats the `com.endeca.eidi.ias.server.port` property as an SSL port. A value of `false` uses HTTP and treats `com.endeca.eidi.ias.server.port` as a non-SSL port. Specify `false` if you enabled redirects from a non-SSL port to an SSL port. Optional.
- `-n` (or `--normalize`) specifies whether to normalize the configuration settings. Specifying this option returns all default configuration settings and their associated default values. Omitting this option returns only user-specified settings. Optional.
- `-p` (or `--port`) specifies the port of the Endeca IAS Service. If the flag is omitted, the default is the value of the `com.endeca.eidi.ias.server.port` property in `commandline.properties`. If the property is not set, the value then defaults to `8510`. Optional.

To get the configuration of a Record Store:

1. Start a command prompt and navigate to `<install path>\IAS\<version>\bin`.
2. Run `recordstore-cmd` and specify options as documented above.

Example of getting the configuration of a Record store

This Windows example gets the configuration for a Record Store named RS1:

```
recordstore-cmd.bat get-configuration -a RS1 -f config.xml -n
```

The command output of the example above is stored in `config.xml` and is also shown here:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<recordStoreConfiguration xmlns="http://recordstore.eidi.endeca.com/">
    <btreePageSize>100</btreePageSize>
    <changePropertyNames/>
    <cleanerInterval>1.0</cleanerInterval>
    <dataDirectory>C:\Oracle\Endeca\IAS\workspace\state\RS1\data</dataDirectory>
    <duplicateRecordCompressionEnabled>false</duplicateRecordCompressionEnabled>
    <generationRetentionTime>168.0</generationRetentionTime>
    <idPropertyName>Endeca.Id</idPropertyName>
    <ignoreInvalidRecords>false</ignoreInvalidRecords>
    <indexWriteFlushInterval>50000</indexWriteFlushInterval>
    <jdbmSettings/>
```

```
    <maxDataFileSize>2147483647</maxDataFileSize>
    <recordCompressionEnabled>false</recordCompressionEnabled>
</recordStoreConfiguration>
```

Getting the ID of the last-committed generation

The `get-last-committed-generation` task retrieves the ID of the last generation that was committed to a Record Store instance.

The syntax for this task is:

```
recordstore-cmd get-last-committed-generation -a RecordStoreInstanceName
[-h HostName] [-l true|false] [-p PortNumber] [-x Id]
```

where:

- `-a` (or `--instanceName`) specifies the name of a Record Store instance. Required.
- `-h` (or `--host`) specifies the host where the Endeca IAS Service is running. If the flag is omitted, the default is the value of the `com.endeca.eidi.ias.server.host` property in `commandline.properties`. If the property is not set, the value then defaults to `localhost`. Optional.
- `-l` (or `--isPortSsl`) specifies whether to communicate with the service using an HTTPS connection. A value of `true` uses HTTPS and treats the `com.endeca.eidi.ias.server.port` property as an SSL port. A value of `false` uses HTTP and treats `com.endeca.eidi.ias.server.port` as a non-SSL port. Specify `false` if you enabled redirects from a non-SSL port to an SSL port. Optional.
- `-p` (or `--port`) specifies the port of the Endeca IAS Service. If the flag is omitted, the default is the value of the `com.endeca.eidi.ias.server.port` property in `commandline.properties`. If the property is not set, the value then defaults to `8510`. Optional.
- `-x` (or `--transaction`) specifies the active transaction ID to use. If you use this option, you must follow it with a `commit-transaction` task to commit the read operation. If this flag is omitted, the operation is done in auto-commit mode. Optional.

Example of getting the last-committed generation ID

The output of this command:

```
recordstore-cmd get-last-committed-generation -a RS1
```

will be similar to this example:

```
The last committed generation: 4
```

The command output shows that Generation 4 was the last generation to be committed to the Record Store.

Getting the last-read generation

The `get-last-read-generation` task retrieves the last-read generation for a given client ID.

Before running this task, make sure to use the `set-last-read-generation` task to set a last-read generation for a specific client ID.

The syntax for this task is:

```
recordstore-cmd get-last-read-generation -a RecordStoreInstanceName
-c ClientId [-h HostName] [-l true|false] [-p PortNumber] [-x Id]
```

where:

- `-a` (or `--instanceName`) specifies the name of a Record Store instance. Required.
- `-c` (or `--client`) specifies a client ID that was previously set with the `set-last-read-generation` task. Required.
- `-h` (or `--host`) specifies the host where the Endeca IAS Service is running. If the flag is omitted, the default is the value of the `com.endeca.eidi.ias.server.host` property in `commandline.properties`. If the property is not set, the value then defaults to `localhost`. Optional.
- `-l` (or `--isPortSsl`) specifies whether to communicate with the service using an HTTPS connection. A value of `true` uses HTTPS and treats the `com.endeca.eidi.ias.server.port` property as an SSL port. A value of `false` uses HTTP and treats `com.endeca.eidi.ias.server.port` as a non-SSL port. Specify `false` if you enabled redirects from a non-SSL port to an SSL port. Optional.
- `-p` (or `--port`) specifies the port of the Endeca IAS Service. If the flag is omitted, the default is the value of the `com.endeca.eidi.ias.server.port` property in `commandline.properties`. If the property is not set, the value then defaults to `8510`. Optional.
- `-x` (or `--transaction`) specifies the active transaction ID to use. If you use this option, you must follow it with a `commit-transaction` task to commit the read operation. If this flag is omitted, the operation is done in auto-commit mode. Optional.

Example of getting the last-read generation

This example gets the last-read generation for the client ID of `rsreader1`:

```
recordstore-cmd get-last-read-generation -a RS1 -c rsreader1
```

The command output is similar to this example:

```
The last read generation id saved for client rsreader1 is: 2
```

In the example, Generation 2 had been previously set as the last read generation for the `rsreader1` client ID.

Getting the ID of the write generation

The `get-write-generation` task returns the ID of the write generation.

The syntax for this task is:

```
recordstore-cmd get-write-generation -a RecordStoreName -x id
[-h HostName] [-l true|false] [-p PortNumber]
```

where:

- `-a` (or `--instanceName`) specifies the name of a Record Store instance. Required.
- `-h` (or `--host`) specifies the host where the Endeca IAS Service is running. If the flag is omitted, the default is the value of the `com.endeca.eidi.ias.server.host` property in `commandline.properties`. If the property is not set, the value then defaults to `localhost`. Optional.
- `-l` (or `--isPortSsl`) specifies whether to communicate with the service using an HTTPS connection. A value of `true` uses HTTPS and treats the `com.endeca.eidi.ias.server.port` property as an SSL port. A value of `false` uses HTTP and treats `com.endeca.eidi.ias.server.port` as a non-SSL port. Specify `false` if you enabled redirects from a non-SSL port to an SSL port. Optional.

- `-p` (or `--port`) specifies the port of the Endeca IAS Service. If the flag is omitted, the default is the value of the `com.endeca.eidi.ias.server.port` property in `commandline.properties`. If the property is not set, the value then defaults to `8510`. Optional.
- `-x` (or `--transaction`) specifies the active transaction ID to use. The transaction must be a `READ_WRITE` type. Required.

Example of getting the write-generation ID

```
recordstore-cmd get-write-generation -a RS1 -x 5
Write generation: 2
```

The output of the `get-write-generation` task shows that Generation 2 is the current write generation.

Listing active transactions

The `list-active-transactions` task lists all the existing active transactions of a specified Record Store instance.

Uncommitted transactions are often the result of an unexpected termination of a crawl or other write operation. In this case, you see an error in the log file that includes the ID of the uncommitted transaction.

The syntax for this task is:

```
recordstore-cmd list-active-transactions  -a RecordStoreInstanceName
[-h HostName] [-l true|false] [-p PortNumber]
```

where:

- `-a` (or `--instanceName`) specifies the name of a Record Store instance. Required.
- `-h` (or `--host`) specifies the host where the Endeca IAS Service is running. If the flag is omitted, the default is the value of the `com.endeca.eidi.ias.server.host` property in `commandline.properties`. If the property is not set, the value then defaults to `localhost`. Optional.
- `-l` (or `--isPortSsl`) specifies whether to communicate with the service using an HTTPS connection. A value of `true` uses HTTPS and treats the `com.endeca.eidi.ias.server.port` property as an SSL port. A value of `false` uses HTTP and treats `com.endeca.eidi.ias.server.port` as a non-SSL port. Specify `false` if you enabled redirects from a non-SSL port to an SSL port. Optional.
- `-p` (or `--port`) specifies the port of the Endeca IAS Service. If the flag is omitted, the default is the value of the `com.endeca.eidi.ias.server.port` property in `commandline.properties`. If the property is not set, the value then defaults to `8510`. Optional.

The task output consists of these fields:

- `ID` - an integer that is the transaction ID.
- `TYPE` - the transaction type, which is `READ` (which supports only Read operations) or `READ_WRITE` (which supports both Read and Write operations).
- `STATUS` - the status of the transaction, which is `ACTIVE` (the transaction is in progress), `COMMITTED` (the transaction has been committed), `COMMIT_FAILED` (the commit task failed for this transaction), or `ROLLED_BACK` (the transaction was rolled back).
- `WRITING_GEN` - either the new generation ID (for `READ_WRITE` types) or `N/A` (for `READ` types, because a new generation is not being written).

- `LAST_COMMITTED` - the generation ID of the last committed generation.

If no transactions are active, this message is displayed:

```
There are no active transactions right now.
```

Example of listing active transactions

If there are active transactions, the output of this command:

```
recordstore-cmd list-active-transactions -a RS1
```

will be similar to this example:

```
ID      TYPE            STATUS  WRITING_GEN     LAST_COMMITTED_GEN
13      READ            ACTIVE  N/A             2
14      READ_WRITE      ACTIVE  3               2
```

Listing generations

The `list-generations` task lists information about the generations that are currently in a Record Store instance.

The syntax for this task is:

```
recordstore-cmd list-generations -a RecordStoreInstanceName
[-h Hostname] [-l true|false] [-p PortNumber]
```

where:

- `-a` (or `--instanceName`) specifies the name of a Record Store instance. Required.
- `-h` (or `--host`) specifies the host where the Endeca IAS Service is running. If the flag is omitted, the default is the value of the `com.endeca.eidi.ias.server.host` property in `commandline.properties`. If the property is not set, the value then defaults to `localhost`. Optional.
- `-l` (or `--isPortSsl`) specifies whether to communicate with the service using an HTTPS connection. A value of `true` uses HTTPS and treats the `com.endeca.eidi.ias.server.port` property as an SSL port. A value of `false` uses HTTP and treats `com.endeca.eidi.ias.server.port` as a non-SSL port. Specify `false` if you enabled redirects from a non-SSL port to an SSL port. Optional.
- `-p` (or `--port`) specifies the port of the Endeca IAS Service. If the flag is omitted, the default is the value of the `com.endeca.eidi.ias.server.port` property in `commandline.properties`. If the property is not set, the value then defaults to `8510`. Optional.

The task output consists of these fields:

- `ID` - an integer that is the generation ID.
- `STATUS` - the status of the generation, which is `STARTED` (the generation is being written to the Record Store instance), `COMPLETED` (the generation has been written and committed to the Record Store instance), or `BEING_CLEANED` (the cleaner is currently cleaning the generation).
- `CREATION TIME` - the date (in YYYYMMDD format) and time (in UTC format) that the generation was created. This value is based on the clock of the machine running the Endeca IAS Service.

If no generations have been written, this message displays:

```
There are no generations in the record store
```

Example of listing generations

If there are generations in the Record Store instance, the output of `list-generations` is similar to this example:

```
recordstore-cmd.bat list-generations -a delimited
FileCrawl
ID      STATUS          CREATION TIME
1       COMPLETED       Tue Apr 16 10:07:28 EDT 2013
2       COMPLETED       Tue Apr 16 10:08:43 EDT 2013
```

The sample shows that there are two generations in the Record Store instance.

Rolling back transactions

The `rollback-transaction` task rolls back an active (uncommitted) transaction for a specified Record Store instance. Once a transaction is rolled back, this cannot be undone.

The syntax for this task is:

```
recordstore-cmd rollback-transaction  -a RecordStoreInstanceName -x Id
[-h HostName] [-l true|false] [-p PortNumber]
```

where:

- `-a` (or `--instanceName`) specifies the name of a Record Store instance. Required.
- `-h` (or `--host`) specifies the host where the Endeca IAS Service is running. If the flag is omitted, the default is the value of the `com.endeca.eidi.ias.server.host` property in `commandline.properties`. If the property is not set, the value then defaults to `localhost`. Optional.
- `-l` (or `--isPortSsl`) specifies whether to communicate with the service using an HTTPS connection. A value of `true` uses HTTPS and treats the `com.endeca.eidi.ias.server.port` property as an SSL port. A value of `false` uses HTTP and treats `com.endeca.eidi.ias.server.port` as a non-SSL port. Specify `false` if you enabled redirects from a non-SSL port to an SSL port. Optional.
- `-p` (or `--port`) specifies the port of the Endeca IAS Service. If the flag is omitted, the default is the value of the `com.endeca.eidi.ias.server.port` property in `commandline.properties`. If the property is not set, the value then defaults to `8510`. Optional.
- `-x` (or `--transaction`) specifies the ID of the transaction that is rolled back. Required.

Note that uncommitted transactions are often the result of an unexpected termination of a crawl. In this case, you see an error in the log file that includes the ID of the uncommitted transaction.

Example of a transaction rollback

This example rolls back the transaction with an ID of 7:

```
recordstore-cmd rollback-transaction -a RS1 -x 7
```

If the command succeeds, it prints the following message:

```
Rolled back transaction: 7
```

If the command fails, it prints the following error message:

```
Failed to roll back transaction: 7
```

Setting the configuration of a Record Store instance

The `set-configuration` task sets configuration settings for a specified Record Store instance.

You can configure a new Record Store instance or reconfigure an existing Record Store instance by specifying an XML configuration file for it.

The syntax for this task is:

```
recordstore-cmd set-configuration -a RecordStoreInstanceName
-f FileName.xml [-h HostName] [-l true|false] [-p PortNumber]
```

where:

- `-a` (or `--instanceName`) specifies the name of a Record Store instance. Required.
- `-f` (or `--file`) specifies the XML file name that contains the configuration settings for a Record Store. Required.
- `-h` (or `--host`) specifies the host where the Endeca IAS Service is running. If the flag is omitted, the default is the value of the `com.endeca.eidi.ias.server.host` property in `commandline.properties`. If the property is not set, the value then defaults to `localhost`. Optional.
- `-l` (or `--isPortSsl`) specifies whether to communicate with the service using an HTTPS connection. A value of `true` uses HTTPS and treats the `com.endeca.eidi.ias.server.port` property as an SSL port. A value of `false` uses HTTP and treats `com.endeca.eidi.ias.server.port` as a non-SSL port. Specify `false` if you enabled redirects from a non-SSL port to an SSL port. Optional.
- `-p` (or `--port`) specifies the port of the Endeca IAS Service. If the flag is omitted, the default is the value of the `com.endeca.eidi.ias.server.port` property in `commandline.properties`. If the property is not set, the value then defaults to `8510`. Optional.

Note: If you set configuration using a file that modifies any of the following properties, the set operation automatically clears all record data in the Record Store instance:

- `btreePageSize`
- `changePropertyNames`
- `idPropertyName`
- `jdbmSettings`
- `recordCompressionEnabled`

To set the configuration of a Record Store:

1. Start a command prompt and navigate to `<install path>\IAS\<version>\bin`.
2. Run `recordstore-cmd` and specify options as documented above.

Example of setting the configuration of a Record Store

This example sets the configuration for a Record Store named RS2:

```
recordstore-cmd.bat set-configuration -a RS2 -f config.xml
```

where the contents of config.xml are as follows:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<recordStoreConfiguration xmlns="http://recordstore.eidi.endeca.com/">
```

```
    <btreePageSize>100</btreePageSize>
    <changePropertyNames/>
    <cleanerInterval>1.0</cleanerInterval>
    <dataDirectory>C:\Oracle\Endeca\IAS\workspace\state\RS1\data</dataDirectory>
    <duplicateRecordCompressionEnabled>false</duplicateRecordCompressionEnabled>
    <generationRetentionTime>168.0</generationRetentionTime>
    <idPropertyName>Endeca.Id</idPropertyName>
    <ignoreInvalidRecords>false</ignoreInvalidRecords>
    <indexWriteFlushInterval>50000</indexWriteFlushInterval>
    <jdbmSettings/>
    <maxDataFileSize>2147483647</maxDataFileSize>
    <recordCompressionEnabled>false</recordCompressionEnabled>
</recordStoreConfiguration>
```

Note: This example deletes all records per the note above.

Setting the last-read generation

The `set-last-read-generation` task sets the last-read generation for a given client ID. This task is the counterpart of `clean-last-read-generation`.

As a result, you are setting the state for that client. This task is mainly used to save the last-read generation for use by a future delta read.

The syntax for this task is:

```
recordstore-cmd set-last-read-generation  -a RecordStoreInstanceName -g GenId
-c ClientId [-h HostName] [-l true|false] [-p PortNumber] [-x Id]
```

where:

- `-a` (or `--instanceName`) specifies the name of a Record Store instance. Required.
- `-g` (or `--generation`) specifies the generation ID to set as the last read for the client. Required.
- `-c` (or `--client`) specifies a string for which the last-read generation will be set. You can use any string for the client ID, as it is used only as an identifier. Required.
- `-h` (or `--host`) specifies the host where the Endeca IAS Service is running. If the flag is omitted, the default is the value of the `com.endeca.eidi.ias.server.host` property in `commandline.properties`. If the property is not set, the value then defaults to `localhost`. Optional.
- `-l` (or `--isPortSsl`) specifies whether to communicate with the service using an HTTPS connection. A value of `true` uses HTTPS and treats the `com.endeca.eidi.ias.server.port` property as an SSL port. A value of `false` uses HTTP and treats `com.endeca.eidi.ias.server.port` as a non-SSL port. Specify `false` if you enabled redirects from a non-SSL port to an SSL port. Optional.
- `-p` (or `--port`) specifies the port of the Endeca IAS Service. If the flag is omitted, the default is the value of the `com.endeca.eidi.ias.server.port` property in `commandline.properties`. If the property is not set, the value then defaults to `8510`. Optional.
- `-x` (or `--transaction`) specifies the active transaction ID to use. If you use this option, you must follow it with a `commit-transaction` task to commit the read operation. If this flag is omitted, the operation is done in auto-commit mode. Optional.

Typically, you use this command so that the Record Store instance can save the state. For example, if you do a baseline-read of Generation 2, you might later come back to the Record Store instance and do a delta read of your last-read generation (in this case it is Generation 2) and the most recently-committed generation. So

you would save the first baseline-read of Generation 2 for the client, for example, `rsreader1`, and then perform a delta read later after getting the last-read generation.

Example of setting the last-read generation

This example sets Generation 2 as the last read for the client ID of `rsreader1`:

```
recordstore-cmd set-last-read-generation -a RS1 -c rsreader1 -g 2
```

If the command succeeds, it prints out this message:

```
Set the last read generation id for client rsreader1 to 2.
```

By using this method, a Record Store Reader component does not need to maintain state in order to use the Record Store.

Starting transactions

The `start-transaction` task begins a Read or Write transaction. Explictly starting and committing transactions is useful if you want to group multiple operations within a single transaction.

If you choose not to use transactions, all read and write operations are performed in auto-commit mode.

The syntax for this task is:

```
recordstore-cmd start-transaction -a RecordStoreInstanceName -t Type
[-h HostName] [-l true|false] [-p PortNumber]
```

where:

- `-a` (or `--instanceName`) specifies the name of a Record Store instance. Required.
- `-h` (or `--host`) specifies the host where the Endeca IAS Service is running. If the flag is omitted, the default is the value of the `com.endeca.eidi.ias.server.host` property in `commandline.properties`. If the property is not set, the value then defaults to `localhost`. Optional.
- `-l` (or `--isPortSsl`) specifies whether to communicate with the service using an HTTPS connection. A value of `true` uses HTTPS and treats the `com.endeca.eidi.ias.server.port` property as an SSL port. A value of `false` uses HTTP and treats `com.endeca.eidi.ias.server.port` as a non-SSL port. Specify `false` if you enabled redirects from a non-SSL port to an SSL port. Optional.
- `-p` (or `--port`) specifies the port of the Endeca IAS Service. If the flag is omitted, the default is the value of the `com.endeca.eidi.ias.server.port` property in `commandline.properties`. If the property is not set, the value then defaults to `8510`. Optional.
- `-t` (or `--transactionType`) specifies the type of transaction to be performed: `READ` (which supports only Read operations) or `READ_WRITE` (which supports both Read and Write operations). Note that the transaction-type arguments are case sensitive. Required.

Note: A `java.lang.IllegalArgumentException` is thrown if the `-t` argument is invalid (such as using lower case).

Example of starting a transaction

This example starts a Read transaction:

```
recordstore-cmd start-transaction -a RS1 -t READ
```

If the command is successful, it prints a message similar to this example:

```
Started transaction: 15
```

Part IV

Administering IAS

Chapter 9

Running IAS Components

This section provides information about how to run the IAS components.

About running IAS components

This topic provides an overview of the recommended way to run IAS components, optional ways to run IAS components, and explains differences between Windows and UNIX platforms.

You run IAS components in any of the following ways:

- In the Endeca IAS Service
- Using command-line utilities
- Programmatically from the IAS APIs (For details, see the *Endeca IAS API Guide*.)

Running IAS components in the Endeca IAS Service

As discussed in the introduction, the Endeca IAS Service is a container process that runs the IAS components such as the IAS Server, the Component Instance Manager, and one or more Record Store instances. In a typical implementation, running the IAS Service is the recommended way to run IAS components.

Running IAS components using the command-line utilities

The Integrator Acquisition System provides several convenience utilities so you can run any component manually from a command prompt, if you choose to. These utilities include the following:

- IAS Server Command-line Utility
- Component Instance Manager Command-line Utility
- Record Store Command-line Utility

Each utility is described in this guide.

Running the Endeca IAS Service on Windows

On Windows, the Endeca IAS Service is automatically started as part of the installation process. Oracle recommends that you start and stop the service from the Microsoft Services console.

You may optionally choose to start the Endeca IAS Service on Windows using the `ias-service.bat` script in `<install path>\IAS\<version>\bin`, and stop it using `ias-service-shutdown.bat`.

Running the Endeca IAS Service on UNIX

On UNIX, you run the Endeca IAS Service using the scripts in `<install path>/IAS/<version>/bin`. You start the service with `ias-service.sh` or via the `inittab`, and stop it with `ias-service-shutdown.sh`. Oracle recommends using the `inittab` in production environments.

Restarting the Endeca IAS Service

On either platform, you can leave the service running as a background process. The only time you must restart the service is if you modify any of the configuration files in `<install path>\IAS\<version>\conf`. For example, you might change the IAS Service logging configuration and therefore have to restart the service.

Running the Endeca IAS Service from the Windows Services console

On Windows, the Endeca IAS Service is registered as a Windows Service and starts automatically when the operating system starts. This is the recommended way of running IAS on Windows.

If you have changed any of the IAS configuration files, you can stop and restart the Endeca IAS service, using the Windows Console, for those changes to take effect.

Note: The IAS Service can be slow to startup if it contains large Record Store instances and has not been cleanly shut down.

To run the Endeca IAS Service on Windows:

1. From the Windows **Start** menu, go to **Control Panel>Administrative Tools>Services**.
2. Locate the Endeca IAS Service from the list and right-click it.
3. From the context menu, select **Stop**, **Restart**, or **Start** as necessary.
4. Exit the Windows Service console.

Starting the Endeca IAS Service from a command prompt

In UNIX environments, you start the Endeca IAS Service from a command prompt. In Windows environments, you can start the service from a command prompt, but it is optional.

Note: The IAS Service can be slow to start up if it contains large Record Store instances and has not been cleanly shut down.

To start the Endeca IAS Service from a command prompt:

1. Open a command prompt and navigate to the `<install path>\IAS\<version>\bin` directory.

2. Run the `ias-service.bat` script (for Windows) or `ias-service.sh` script (for UNIX).
   For example, in a default installation on a Windows machine, the command is as follows:

   ```
   C:\Oracle\Endeca\IAS\3.0.0\bin>ias-service
   ```

   This command starts the IAS Service on the default port 8510 with a workspace directory of `<install path>\IAS\workspace`.

   Note that you do not see any startup messages. All messages are sent to the IAS Service log (by default, `workspace\logs\ias-service.log`). However, if there is an error in setting up logging, all messages are sent to the console.

3. Verify that the server is running by opening a Web browser and entering a URL with the IAS Service host and port number followed by `ias/?wsdl`. For example, in a default installation:

   ```
   http://localhost:8510/ias/?wsdl
   ```

   You see the IasCrawlerService WSDL in the Web browser window, which indicates that the Endeca IAS Service is running.

Command-line flags to IAS Service

The Endeca IAS Service startup script has an optional Java Virtual Machine (-JVM) flag.

Note: Flag names are case sensitive.

ias-service Flag	Flag Argument
`-JVM`	Allows arguments on the command line to be passed to the JVM. If this flag is used, any arguments after it are passed to the IAS Service and any arguments afterwards are appended to those passed to the JVM. Note that on Windows machines, the flag parameters should be quoted if they contain equal signs. Optional.

Specifying JVM arguments

To pass arguments to the JVM, you can use the `-JVM` script flag. For example, assume you want to override the default maximum heap size setting of 1024 MB with a setting of 2048 MB. The command line is as follows:

```
ias-service -JVM -Xmx2048m
```

Keep in mind that this flag must be the last flag on the command line, because any arguments that follow it are appended to those passed to the JVM.

Stopping the Endeca IAS Service from a command prompt

In UNIX environments, you stop the Endeca IAS Service from a command prompt. In Windows environments, you can optionally stop the service from a command prompt, or use the Windows Services console (recommended).

Note: If you start the service using the command prompt, you should also stop the service using the command prompt.

To stop the Endeca IAS Service:

1. Open a command prompt and navigate to the `bin` directory.

   In a default installation on Windows, this is `C:\Oracle\Endeca\IAS\<version>\bin`.

2. Run the `ias-service-shutdown.bat` script (for Windows) or `ias-service-shutdown.sh` script (for UNIX).

   For example, in a default installation on a Windows machine, the command is as follows:

   ```
   C:\Oracle\Endeca\IAS\3.0.0\bin>ias-service-shutdown
   ```

Chapter 10

Backing up and Restoring IAS

This section describes how to back up and restore IAS state, crawl configurations, and Record Store instance data.

Coordinating backups and restore operations

Online backups are often done more frequently than offline backups. For example, you might perform a full offline backup once a week, and perform smaller online backups on a daily basis. So when you restore the backup, you would have to first restore the weekly offline backup and then the series of daily online backups.

Online backup and restore operations

The administration tasks in this section can be performed while the Endeca IAS Service is running. The tasks are generally more focused and specific than the tasks you can perform while the Endeca IAS Service is offline. And there are some elements of IAS that you cannot back up online, for example, you cannot back up crawl histories while the Endeca IAS Service is running.

Backing up crawl configurations

You back up crawl configurations using the IAS Server Command-line Utility.

To back up crawl configurations:

1. Start a command prompt and navigate to `<install path>\IAS\<version>\bin` (for Windows), or `<install path>/IAS/<version>/bin` (for UNIX).
2. Type `ias-cmd.bat` (for Windows), or `ias-cmd.sh` (for UNIX) and specify the `getAllCrawls` task with the `-f` (or `--file`) flag and the name of the XML file to write the crawl configurations to.
   For example:

   ```
   ias-cmd getAllCrawls -f C:\tmp\backupconfig.xml
   ```

   Note that password configuration properties are not stored in the crawl configuration.

Backing up the last generation of Endeca records

This procedure describes how to back up the last generation of Endeca records in a Record Store instance and back up the corresponding configuration for the Record Store instance. This task does not describe backing up multiple generations or deltas between generations.

To back up the last generation of Endeca records:

1. Start a command prompt and navigate to `<install path>\IAS\<version>\bin` (for Windows), or `<install path>/IAS/<version>/bin` (for UNIX).
2. To list the available Record Store instances in IAS, type `component-manager-cmd.bat` (for Windows), or `component-manager-cmd.sh` (for UNIX) and specify the `list-components` task.

   For example:

   ```
   component-manager-cmd.bat list-components
   NAME            TYPE            STATUS
   Test         RecordStore     RUNNING
   ```

3. From the list, identify the Record Store instance that contains the generation of records you want to back up.
4. Type `recordstore-cmd.bat` (for Windows), or `recordstore-cmd.sh` (for UNIX) and specify the `read-baseline` task with the `-a` (or `--instanceName`) flag and the name of a Record Store instance and also the `-f` (or `--file`) flag and the pathname of the file to which the Endeca records will be output.

   Valid extensions for the file are `.xml` (for an XML format) and `.bin` (for a binary format); the file can also have an additional, optional `.gz` extension if it is a compressed file. Oracle recommends using `.bin.gz` because it is the most compact format.

   For example:

   ```
   recordstore-cmd.bat read-baseline -a Test -f C:\tmp\RSIbackup.xml
   ```

   The `read-baseline` operation writes the last generation of Endeca records. It does not write all generations.
5. To back up the configuration for a Record Store instance, type `recordstore-cmd.bat` (for Windows), or `recordstore-cmd.sh` (for UNIX) and specify the `get-configuration` task with the `-a` (or `--instanceName`) flag and the name of a Record Store instance and also the `-f` (or `--file`) flag and the XML file name where you want to save the configuration settings.

   For example:

   ```
   recordstore-cmd.bat get-configuration -a Test -f C:\tmp\RSIbackup_configfile.xml
   ```

Restoring crawl configurations

You restore crawl configurations using the IAS Server Command-line Utility.

To restore crawl configurations:

1. Start a command prompt and navigate to `<install path>\IAS\<version>\bin` (for Windows), or `<install path>/IAS/<version>/bin` (for UNIX).

2. If you are restoring into a system that has an older version of the crawl configuration, type `ias-cmd.bat` (for Windows), or `ias-cmd.sh` (for UNIX) and specify the `updateCrawls` task with the `-f` (or `--file`) flag and the name of the XML file that contains crawl configurations.

   For example:

   ```
   ias-cmd updateCrawls -f C:\tmp\backupconfig.xml
   Updated crawl Test
   ```

3. If you are restoring into a system that does not have the crawl configuration, type `ias-cmd.bat` (for Windows), or `ias-cmd.sh` (for UNIX) and specify the `createCrawls` task with the `-f` (or `--file`) flag and the name of the XML file that contains crawl configurations.

   For example:

   ```
   ias-cmd createCrawls -f C:\tmp\backupconfig.xml
   Updated crawl Test
   ```

Restoring the last generation of Endeca records

This task describes restoring one generation of baseline data into a Record Store instance and restoring the corresponding configuration file for the Record Store instance. This task does not describe restoring multiple generations or deltas between generations.

To restore the last generation of Endeca records:

1. Start a command prompt and navigate to `<install path>\IAS\<version>\bin` (for Windows), or `<install path>/IAS/<version>/bin` (for UNIX).
2. Create a new empty Record Store instance by typing `component-manager-cmd.bat` (for Windows), or `component-manager-cmd.sh` (for UNIX) and specify the `create-component` task with the `-t` option with an argument of `RecordStore`, and the`-n` option with a Record Store instance name of your choice.

   You need to repeat this step if your crawl configuration contains multiple Record Store instances. Also, you should ensure that the name of each Record Store instance coordinates with the `crawlId`. For example, if you have a `crawlId` of `Test`, you create a Record Store instance named `Test`.

   For example:

   ```
   component-manager-cmd.bat create-component
    -n Test -t RecordStore
   ```

3. Restore the configuration file for a Record Store instance by typing `recordstore-cmd.bat` (for Windows), or `recordstore-cmd.sh` (for UNIX) and specify the `set-configuration` task with the `-a` (or `--instanceName`) flag and the name of a Record Store instance and also the `-f` (or `--file`) flag and the XML file name that contains the configuration settings.

   For example:

   ```
   recordstore-cmd.bat set-configuration -a Test -f C:\tmp\RSIbackup_configfile.xml
   Successfully set recordstore configuration.
   ```

4. Write the data into the Record Store instance by typing `recordstore-cmd.bat` (for Windows), or `recordstore-cmd.sh` (for UNIX) and specify the `write` task with the `-a` (or `--instanceName`)

flag and the name of a Record Store instance, and also the `-f` (or `--file`) flag and the pathname of the file that contains the Endeca records.

For example:

```
recordstore-cmd.bat write -a Test -f C:\tmp\RSIbackup.xml
Wrote 2190 records.
```

Offline backup and restore operations

The administration tasks in this section can only be performed while the Endeca IAS Service is stopped. Once you shut down the service, you can back up IAS to preserve nearly all of its state.

The IAS state includes:

- Record Store data and configuration
- Crawl configurations
- Crawl history

The IAS state does not include:

- Alternate data directories for Record Store instance data (configured via the `dataDirectory` property).
- State information for custom data sources or manipulators that write state to other locations. (Extensions developed using the IAS Extension API can write to any location a developer choses.)

Backing up IAS state

This task describes how to back up IAS state. IAS stores its state in `<install path>\IAS\workspace\state`.

To back up IAS state:

1. Stop the Endeca IAS Service.
2. On the machine running the IAS Service, navigate to `<install path>\IAS\workspace\state`.
3. Copy the `state` directory to a location outside the IAS installation.
4. Restart the Endeca IAS Service.

Restoring IAS state

This task describes how to restore IAS state information.

To restore IAS state:

1. Stop the Endeca IAS Service.
2. Locate the `ias` directory that you previously backed up. This is typically in a location outside the IAS installation.
3. Navigate to `<install path>\IAS\Workspace\state`.
4. Copy the `state` directory that you previously backed up into `<install path>\IAS\Workspace`.

5. Restart the Endeca IAS Service.

Chapter 11

Configuring Logging

This section describes how to configure logging for IAS.

Configuring logging for IAS components and command-line utilities

You can change the default logging configuration of the Endeca IAS Service and any IAS components you run from the command line.

Log location and rolling

IAS writes log files to the `<install path>\IAS\workspace\logs` directory. If desired, you can reconfigure IAS to write log files to another location.

IAS rolls a log file once it exceeds 100MB, and the IAS Service keeps 10 backups of its log.

Note: If you delete the log file, the Endeca IAS Service recreates the log only when you restart the service. So it is possible to delete the log, run a crawl, and not have logging information if you did not restart the service.

Configuration files for IAS components

The `<install path>\IAS\workspace\conf` directory contains the following logging configuration files:

- `ias-cmd.log4j.properties` - configures logging for the IAS Server Command-line Utility
- `ias-service-log4j.properties` - configures logging for the Endeca IAS Service
- `component-manager-cmd.log4j.properties` - configures logging for the Component Instance Manager Command-line Utility
- `recordstore-cmd.log4j.properties` - configures logging for the Record Store Command-line Utility

Logging options and levels

You can re-configure log locations, log file size, log file encoding, log pattern, and logging message levels.

Logging levels can be set to any of the following:

- `DEBUG` designates fine-grained informational events that are most useful to debug Record Store problems.
- `INFO` designates informational messages that highlight the progress of Record Store operations at a coarse-grained level.
- `WARN` designates potentially harmful situations.
- `ERROR` designates error events that might still allow the Record Store to continue running.
- `FATAL` designates very severe error events that will presumably lead the Record Store to abort.
- `OFF` has the highest possible rank and is intended to turn off logging.

These levels allow you to monitor events of interest at the appropriate granularity. When you are initially setting up your Record Store implementation, you might want to use the `DEBUG` level to get all messages, and change to a less verbose level in production.

Log file encoding

IAS produces log files encoded as UTF-8.

Setting log properties to troubleshoot CMS crawls issues

You can set logging properties that may help determine the causes of connection, authentication, and request/response time issues between IAS and a CMS repository.

To set log properties to troubleshoot CMS crawls issues:

1. In a text editor, open `ias-service-log4j.properties`.
2. Add the following lines to the file:

```
log4j.logger.org.apache.axis.client=DEBUG
log4j.logger.httpclient.wire=DEBUG
log4j.logger.org.apache.commons.httpclient=DEBUG
```

Excluding failed records from the IAS Service log file

If a record fails during a crawl, the IAS Server discards the failed record and writes a truncated version of the record to the `ias-service.log` file. If you do not want the IAS Server to write any information about failed records to the log file, you can disable logging for failed records by un-commenting a setting in the `ias-service-log4j.properties` file.

To exclude failed records from the IAS Service log file:

1. Stop the Endeca IAS Service.
2. Navigate to the `<install path>\IAS\workspace\conf` directory.
3. In a text editor, open `ias-service-log4j.properties`.
4. Uncomment the line containing the `log4j.logger.com.endeca.eidi.executor.ErrorChannelImpl` setting.
5. Save and close the `ias-service-log4j.properties` file.

6. Restart the Endeca IAS Service.

Once you un-comment the setting, the IAS Server does not write any information about failed records to the log file. However, failed records are still counted as metrics under FAILED_RECORDS.

Enabling log timing information for crawl processing steps

You can enable a logging setting in `ias-service-log4j.properties` that instructs IAS to write log timing information for each processing step of a crawl. This additional logging information is especially useful for troubleshooting performance issues.

1. In a text editor, open `<install path>\IAS\workspace\conf\ias-service-log4j.properties`.
2. Un-comment the following line in the file:
   ```
   log4j.logger.com.endeca.eidi.executor.ProcessorTaskTiming=DEBUG
   ```
3. Save and close the file.
4. Restart the Endeca IAS Service.

The next time you run a crawl you will get additional logging information similar to the following:

```
Processor Task Timing

IncrementalDataSourceProcessor-414611937: (Hits=1, Value=8542.280 ms, Time=15:11:37,489)
MdexOutputSink-1898864883(processRecord): (Total=7413.427 ms, Avg=0.835 ms, Hits=8877, StdDev
=9.526 ms, Min=0.001 ms, Max=659.003 ms, FirstTimed=15:11:36,802, LastTimed=15:11:40,078)
SplittingFilterProcessor-1235020019(processRecord): (Total=3250.518 ms, Avg=0.366 ms, Hits
=8877, StdDev=0.711 ms, Min=0.010 ms, Max=26.920 ms, FirstTimed=15:11:29,626, LastTimed=15:11:37,457)
ArchiveExpandProcessor-1134860470(processRecord): (Total=2104.446 ms, Avg=0.237 ms, Hits=8877, StdDev
=0.843 ms, Min=0.004 ms, Max=44.004 ms, FirstTimed=15:11:29,595, LastTimed=15:11:37,457)
PropertyRemover-92265517(processRecord): (Total=1849.963 ms, Avg=0.208 ms, Hits=8877, StdDev
=0.716 ms, Min=0.003 ms, Max=27.465 ms, FirstTimed=15:11:29,595, LastTimed=15:11:37,457)
MdexOutputSink-1898864883(notifyInputClosed): (Total=598.802 ms, Avg=299.401 ms, Hits=2, StdDev
=391.645 ms, Min=22.466 ms, Max=576.336 ms, FirstTimed=15:11:38,206, LastTimed=15:11:40,094)
PropertyRemover-92265517(notifyInputClosed): (Hits=1, Value=0.544 ms, Time=15:11:37,489)
ArchiveExpandProcessor-1134860470(notifyInputClosed): (Total=0.330 ms, Avg=0.165 ms, Hits=2, StdDev
=0.141 ms, Min=0.065 ms, Max=0.265 ms, FirstTimed=15:11:37,489, LastTimed=15:11:37,489)
SplittingFilterProcessor-1235020019(notifyInputClosed): (Hits=1, Value=0.012 ms, Time=15:11:37,489)
```

Examining the Endeca IAS Service log

The Endeca IAS Service logs messages for all IAS components and crawls in the `ias-service.log` file.

Location of the IAS Service log

The Endeca IAS Server has one (and only one) log, regardless of how many crawls have been configured. The log is named `ias-service.log` and is located in the `logs` directory in the IAS workspace directory. If you are using the default workspace directory name, the pathname of the log file is similar to this:

```
C:\Oracle\Endeca\IAS\workspace\logs\ias-service.log
```

Format of log entries

The log contains two types of log entries:

- IAS component log entries, which are entries that pertain to starting and stopping IAS components.
- Crawl log entries, which are entries that pertain to a specific crawl.

By default, crawl log entries have the format:

```
yyyy-MM-dd HH:mm:ss,SSS logLevel [component] [thread name] class: <message>
```

where:

- *yy-MM-dd HH-mm-ss* is the timestamp of the entry. You can change the format by editing the `ias-server.log4j.properties` file.
- *logLevel* is the log level of the entry, such as INFO or FATAL.
- *component* is ias (for the crawl manager), ComponentInstanceManager, or instance name for Record Stores.
- *thread name* is the name of the processing thread for the message.
- *message* is the message returned by a IAS Server module.

Enabling crawl statistics

If a crawl log level is set to INFO, TRACE, or DEBUG, the crawl statistics are entered as INFO entries in the log when the crawl finishes, as in this example (timestamps and log levels are omitted for ease of reading):

```
Crawl Mode = FULL_CRAWL (MetricsReport)
Crawl Stop Cause = Completed (MetricsReport)
Directories Filtered from Archives = 0 (MetricsReport)
Directories Filtered = 0 (MetricsReport)
Total Records Output = 423 (MetricsReport)
Files Filtered from Archives = 124 (MetricsReport)
Directories Crawled Not from Archives = 55 (MetricsReport)
Documents Unsuccessfully Converted = 9 (MetricsReport)
Files Crawled from Archives = 65 (MetricsReport)
Files Crawled Not from Archives = 285 (MetricsReport)
Delete Records Output = 0 (MetricsReport)
Files Filtered Not from Archives = 51 (MetricsReport)
Directories Crawled = 73 (MetricsReport)
Directories Filtered Not from Archives = 0 (MetricsReport)
Documents Converted = 333 (MetricsReport)
Files Crawled = 350 (MetricsReport)
Documents Converted After Retry = 0 (MetricsReport)
New or Updated Records Output = 423 (MetricsReport)
Directories Crawled from Archives = 18 (MetricsReport)
Files Filtered = 175 (MetricsReport)
Crawl Seconds = 71 (MetricsReport)
Start Time = 5/23/08 9:23:59 AM EDT (MetricsReport)
End Time = 5/23/08 9:25:10 AM EDT (MetricsReport)
```

Note that for incremental crawls, the `Delete Records Output` statistic is also included and indicates how many files were deleted from the previous crawl. An Endeca record is created for each deleted file; the record will have the `Endeca.Action` property set to DELETE.

The `Crawl Stop Cause` statistic has one of the following values:

- Completed
- Failed

- Aborted

If a crawl fails, the `Crawl Failure Reason` statistic provides a message from the IAS Server explaining the failure.

Keep in mind that if the log is too verbose (thus making it more difficult to find errors), you can change the log level of the crawl. The default log level is INFO.

The IAS logging configuration file is `ias-service-log4j.properties` and is located in the `<install path>\IAS\workspace\conf` directory. You can also change the log level on a per-crawl basis using the IAS API or the IAS command-line utilities.

Chapter 12

Tips and Troubleshooting IAS

This section provides tips and miscellaneous troubleshooting information about IAS.

Fixing crawl performance issues

This topic lists performance issues you many encounter when running a IAS crawl and provides ways to address the issues.

Periodic crawl performance problems caused by defragmenting the crawl history database

IAS has an embedded database that stores crawl history. With large crawls, the database can grow until it approaches its maximum file size limit. However, before the database reaches its maximum size, IAS defragments it, in order to reduce the overall file size of the database. Any crawls that are running will slow down while the defragmentation process is running.

If you notice a crawl that is taking a long time, you can determine whether a defragmentation process is causing the issue by checking the IAS service log for the following error:

`com.endeca.eidi.EidiRuntimeException: java.sql.SQLException: Data File size limit is reached`.

If you see this error frequently, you can work with Oracle Customer Support to adjust the maximum file size of the crawl history database and adjust the frequency at which IAS runs the defragmentation process. For details, contact Oracle Customer Support.

Modifying the IAS Service temporary directory

By default, the Endeca IAS Service temporary directory is set to `<install path>\IAS\workspace\temp` (on Windows) and `<install path>/IAS/workspace/temp` (on UNIX). If necessary, you can modify this path by changing the `java.io.tmpdir` system property in the Endeca IAS Service script.

To modify the IAS Service temporary directory:

1. Stop the Endeca IAS Service.
2. Navigate to `<install path>\IAS\version\bin`.

3. If you are running the Endeca IAS Service manually, open `ias-service` (either .bat or .sh depending on your platform) in a text editor.
4. If you are running the Endeca IAS Service automatically as a Windows service, open `ias-service-wrapper.conf` in a text editor.
5. Locate the `Djava.io.tmpdir` argument and modify the value of the path as necessary.
6. Save and close the file.
7. Restart the Endeca IAS Service.

Responding to a "Too many open files" error

On UNIX, you may get a “Too many open files” error if you are crawling several data sources simultaneously.

The relevant line in the error’s stack trace is the following:

```
Caused by: java.io.FileNotFoundException: /localdisk/jsmith/endeca/IAS/workspace/state
/test_data_multiseeds/data/dictionary/seg0/c3a1.dat (Too many open files)
```

The error occurs because the operating system has reached the per-process limit for the number of files the process can have open at once.

To resolve this problem, you can increase the number of file handles available. For more information about how to increase the number of available file handles, refer to the documentation for your operating system.

Note: There is no single recommended range of file handles values that will fit all situations. File/socket requirements can depend on a number of metrics, such as processes managed, nodes, files transferred, and system status queries. Therefore, determining a new limit experimentally, through trial and error, is the simplest resolution.

Setting the group entry size

You can change the group entry size default setting.

On UNIX systems, IAS relies on the group and passwd databases to generate native properties for files. Because there is no limit to the size of the entries in these databases, the default sizes may be too large for some systems.

For example, if the size of a group entry is too large, the following message is written to the log:

```
The group's entry in the group database is too large,
consider setting the com.endeca.eidi.group.size property.
```

You change group entry size by using the Java `-D` option as a parameter to the Java Virtual Machine (JVM), as follows:

```
-Dcom.endeca.eidi.group.size=2048
```

Note that the `2048` parameter is in bytes.

To pass this parameter to the JVM, use the `-JVM` flag when you run the startup script.

Keep in mind that the `-JVM` flag must be the last flag on the command line.

This type of error is more likely to occur with entries in the group database, rather than the passwd database. If, however, a crawl encounters problems with the passwd database, there is also a passwd entry property:

```
com.endeca.eidi.passwd.size
```

Appendix A

File Formats Supported by the IAS Document Conversion Module

This section lists the binary file formats that the IAS Document Conversion Module can convert to text during a crawl. The IAS Document Conversion Module is installed by default as part of the IAS installation.

Archive formats

The following table lists supported archive formats:

Format	Version (if applicable)
7z (BZIP2 and split archives not supported)	
7z Self Extracting exe (BZIP2 and split archives not supported)	
LZA Self Extracting Compress	
LZH Compress	
Microsoft Binder	95, 97
RAR	1.5, 2.0, 2.9

Format	Version (if applicable)
Self-extracting .exe	
UNIX Compress	
UNIX GZip	
UNIX TAR	
Uuencode	
ZIP	PKZip
ZIP	WinZip

Database formats

The following table lists supported database formats:

Format	Version
DataEase	4.x
DBase	III, IV, and V
First Choice DB	Through 3.0
Framework DB	3.0
Microsoft Access	1.0, 2.0
Microsoft Access Report Snapshot (File ID only)	2000 - 2003
Microsoft Works DB for DOS	1.0, 2.0
Microsoft Works DB for Macintosh	2.0
Microsoft Works DB for Windows	3.0, 4.0
Paradox (DOS)	2.0 - 4.0
Paradox (Windows)	1.0
Q & A	Through 2.0
R:Base	R:Base 5000 and R:Base System V

Format	Version
Reflex	2.0
SmartWare II	1.02

E-mail formats

The following table lists supported e-mail formats:

Format	Version
Apple Mail Message (EMLX)	2.0
Encoded mail messages	MHT
Encoded mail messages	Multi Part Alternative
Encoded mail messages	Multi Part Digest
Encoded mail messages	Multi Part Mixed
Encoded mail messages	Multi Part News Group
Encoded mail messages	Multi Part Signed
Encoded mail messages	TNEF
IBM Lotus Notes Domino XML Language DXL	8.5
IBM Lotus Notes NSF (File ID only)	7.x, 8.x
IBM Lotus Notes NSF (Windows, Linux x86-32 and Oracle Solaris 32-bit only with Notes Client or Domino Server)	8.x
MBOX Mailbox	RFC 822
Microsoft Outlook MSG	97 - 2007
Microsoft Outlook Express (EML)	
Microsoft Outlook Forms Template (OFT)	97 - 2007
Microsoft Outlook OST	97 - 2007

Format	Version
Microsoft Outlook PST	97 - 2007
Microsoft Outlook PST (Mac)	2001

Multimedia formats

The following table lists supported e-mail formats:

Format	Version
AVI (Metadata extraction only)	
Flash (text extraction only)	6.x, 7.x, Lite
Flash (File ID only)	9, 10
Real Media - (File ID only)	
MP3 (ID3 metadata only)	
MPEG-1 Audio layer 3 V ID3 v1 (File ID only)	
MPEG-1 Audio layer 3 V ID3 v2 (File ID only)	
MPEG-1 Video V 2 (File ID only)	
MPEG-1 Video V 3 (File ID only)	
MPEG-2 Audio (File ID only)	
MPEG-4 (Metadata extraction only)	
MPEG-7 (Metadata extraction only)	
QuickTime (Metadata extraction only)	
Windows Media ASF (Metadata extraction only)	
Windows Media DVR-MS (Metadata extraction only)	

Format	Version
Windows Media Audio WMA (Metadata extraction only)	
Windows Media Playlist (File ID only)	
Windows Media Video WMV (Metadata extraction only)	
WAV (Metadata extraction only)	

Other formats

The following table lists other supported formats:

Format	Version (if applicable)
AOL Messenger (File ID only)	7.3
Microsoft InfoPath (file ID only)	2007
Microsoft Live Messenger (via XML filter)	10.0
Microsoft OneNote (file ID only)	2007
Microsoft Project (table view only)	98 - 2003
Microsoft Project (table view only)	2007 - 2010
Microsoft Windows Compiled Help (File ID only)	.chm
Microsoft Windows DLL	
Microsoft Windows Executable	
Microsoft Windows Explorer Command (File ID only)	.scf

Format	Version (if applicable)
Microsoft Windows Help (File ID only)	.hlp
Microsoft Windows Shortcut (File ID only)	.lnk
Trillian Text Log File (via text filter)	4.2
Trillian XML Log File (File ID only)	4.2
TrueType Font (File ID only)	ttf, ttc
vCalendar	2.1
vCard	2.1
Yahoo! Messenger	6.x - 8.0

Presentation formats

The following table lists supported presentation formats:

Format	Version (if applicable)
Corel Presentations	6.0 - X3
Harvard Graphics (DOS)	3.0
IBM Lotus Symphony Presentations	1.x
Kingsoft WPS Presentation	2010
Lotus Freelance	1.0 - Millennium 9.6
Lotus Freelance (OS/3)	2.0
Lotus Freelance for Windows	95, 97
Microsoft PowerPoint for Macintosh	4.0 - 2008
Microsoft PowerPoint for Windows	3.0 - 2010
Microsoft PowerPoint for Windows Slideshow	2007 - 2010

Format	Version (if applicable)
Microsoft PowerPoint for Windows Template	2007 - 2010
Novell Presentations	3.0, 7.0
OpenOffice Impress	1.1, 3.0
Oracle Open Office Impress	3.x
StarOffice Impress	5.2 - 9.0
WordPerfect Presentations	5.1 - X4

Raster image formats

The following table lists supported raster image formats:

Format	Version
CALS Raster (GP4)	Type I and Type II
Computer Graphics Metafile	ANSI, CALS, NIST
Encapsulated PostScript (EPS)	TIFF header only
GEM Image (Bitmap)	
Graphics Interchange Format (GIF)	
IBM Graphics Data Format (GDF)	1.0
IBM Picture Interchange Format (PIF)	1.0
JBIG2	graphic embeddings in PDF files
JFIF (JPEG not in TIFF format)	
JPEG	
JPEG 2000	JP2
Kodak Flash Pix	
Kodak Photo CD	1.0
Lotus PIC	

Format	Version
Lotus Snapshot	
Macintosh PICT1 and PICT2	BMP only
MacPaint	
Microsoft Windows Bitmap	
Microsoft Windows Cursor	
Microsoft Windows Icon	
OS/2 Bitmap	
OS/2 Warp Bitmap	
Paint Shop Pro (Win32 only)	5.0, 6.0
PC Paintbrush (PCX)	
PC Paintbrush DCX (multi-page PCX)	
Portable Bitmap (PBM)	
Portable Graymap (PGM)	
Portable Network Graphics (PNG)	
Portable Pixmap (PPM)	
Progressive JPEG	
StarOffice Draw	6.x - 9.0
Sun Raster	
TIFF	Group 5 and Group 6
TIFF CCITT Fax	Group 3 and Group 4
Truevision TGA (Targa)	2.0
WBMP wireless graphics format	
Word Perfect Graphics	1.0
X-Windows Bitmap	x10 compatible
X-Windows Dump	x10 compatible

Format	Version
X-Windows Pixmap	x10 compatible
WordPerfect Graphics	2.0, 7.0, 8.0, 9.0, 10.0

Spreadsheet formats

The following table lists supported spreadsheet formats:

Format	Version
Enable Spreadsheet	3.0 - 4.5
First Choice SS	Through 3.0
Framework SS	3.0
IBM Lotus Symphony Spreadsheets	1.x
Kingsoft WPS Spreadsheets	2010
Lotus 1-2-3	Through Millennium 9.6
Lotus 1-2-3 Charts (DOS and Windows)	Through 5.0
Lotus 1-2-3 (OS/2)	2.0
Microsoft Excel Charts	2.x - 2007
Microsoft Excel for Macintosh	98 - 2008
Microsoft Excel for Windows	3.0 - 2010
Microsoft Excel for Windows (xslb)	2007 - 2010 Binary
Microsoft Multiplan	4.0
Microsoft SS Works for DOS	2.0
Microsoft Works for Macintosh	2.0
Microsoft SS Works for Windows	3.0, 4.0
Novell PerfectWorks	2.0
OpenOffice Calc	1.1 - 3.0

Format	Version
Oracle Open Office Calc	3.x
PFS: Professional Plan	1.0
Quattro for DOS	Through 5.0
QuattroPro for Windows	Through X4
SmartWare Spreadsheet	
SmartWare II SS	1.02
StarOffice Calc	5.2 - 9.0
SuperCalc	5.0
Symphony	Through 2.0
VP Planner	1.0

Text and markup formats

The following table lists supported text and markup formats:

Notes:

- IAS supports converting XML content contained in both PCDATA and CDATA elements.
- In the case of XHTML, "file ID only" means that the conversion process produces an Endeca property for the file format type but nothing else.

Format	Version (if applicable)
ANSI Text	7 bit and 8 bit
ASCII Text	7 bit and 8 bit
DOS character set	
EBCDIC	
HTML (CSS rendering not supported)	1.0 - 4.0
IBM DCA/RFT	
Macintosh character set	
Rich Text Format (RTF)	

Format	Version (if applicable)
Unicode Text	3.0, 4.0
UTF-8	
Wireless Markup Language	1.0
XML	text only
XHTML (file ID only)	1.0

Vector image formats

The following table lists supported vector image formats:

Format	Version (if applicable)
Adobe Illustrator	4.0 - 7.0, 9.0
Adobe Illustrator (XMP only)	11 - 13 (CS 1 - 3)
Adobe InDesign (XMP only)	3.0 - 5.0 (CS 1 - 3)
Adobe InDesign Interchange (XMP only)	
Adobe Photoshop (XMP only)	8.0 -10.0 (CS 1 - 3)
Adobe PDF	1.0 - 1.7 (Acrobat 1 - 9)
Adobe PDF Package	1.7 (Acrobat 8 - 9)
Adobe PDF Portfolio	1.7 (Acrobat 8 - 9)
Adobe Photoshop	4.0
Ami Draw	SDW
AutoCAD Drawing	2.5, 2.6
AutoCAD Drawing	9.0 - 14.0
AutoCAD Drawing	2000i - 2010
AutoShade Rendering	2.0
Corel Draw	2.0 - 9.0

Format	Version (if applicable)
Corel Draw Clipart	5.0, 7.0
Enhanced Metafile (EMF)	
Escher graphics	
FrameMaker Vector and Raster Graphics (FMV)	3.0 - 5.0
Gem File (Vector)	
Harvard Graphics Chart (DOS)	2.0 - 3.0
Harvard Graphics for Windows	
HP Graphics Language	2.0
Initial Graphics Exchange Specification (IGES) Drawing	5.1 - 5.3
Micrografx Designer	Through 3.1
Micrografx Designer	6.0
Micrografx Draw	Through 4.0
Microsoft XPS (Text only)	
Novell PerfectWorks Draw	2.0
OpenOffice Draw	1.1 - 3.0
Oracle Open Office Draw	3.x
Visio (Page Preview mode only WMF/EMF)	4
Visio	5.0 - 2007
Visio XML VSX (File ID only)	2007
Windows Metafile	

Notes on Adobe PDF text extraction

The IAS Document Conversion Module works as follows when processing Adobe PDF files with security settings:

- The IAS Document Conversion Module will respect the no-copy option of a PDF. That is, if a PDF publishing application has a no-copy option (which prohibits the copying or extraction of text within the PDF), the Document Conversion Module will not extract text from that PDF.
- The IAS Document Conversion Module does not support text extraction from password-protected files.
- The IAS Document Conversion Module does not support text extraction from PDFs with encrypted content.

To extract the text from these types of PDFs, you must re-create them without setting the appropriate security option.

In addition, text added with the Sticky Note tool is not extracted.

Word processing formats

The following table lists supported word processing formats:

Format	Version (if applicable)
Adobe FrameMaker (MIF)	Versions 3.0 - 6.0
Adobe Illustrator Postscript	Level 2
Ami	
Ami Pro for OS2	
Ami Pro for Windows	2.0, 3.0
DEC DX	Through 4.0
DEC DX Plus	4.0, 4.1
Enable Word Processor	3.0 - 4.5
First Choice WP	1.0, 3.0
Framework WP	3.0
Hangul	97 - 2007
IBM DCA/FFT	
IBM DisplayWrite	2.0 - 5.0
IBM Writing Assistant	1.01

Format	Version (if applicable)
Ichitaro	5.0, 6.0, 8.0 - 13.0, 2004
JustWrite	Through 3.0
Kingsoft WPS Writer	2010
Legacy	1.1
Lotus Manuscript	Through 2.0
Lotus WordPro	9.7, 96, - Millennium 9.6
Lotus WordPro (non-Win32)	97 - Millennium 9.6
MacWrite II	1.1
Mass 11	All versions through 8.0
Microsoft Publisher (File ID only)	2003 - 2007
Microsoft Word for DOS	4.0 - 6.0
Microsoft Word for Macintosh	4.0 - 6.0, 98 - 2008
Microsoft Word for Windows	1.0 - 2007
Microsoft Word for Windows	98-J
Microsoft WordPad	
Microsoft Works WP for DOS	2.0
Microsoft Works WP for Macintosh	2.0
Microsoft Works WP for Windows	3.0, 4.0
Microsoft Write for Windows	1.0 - 3.0
MultiMate	Through 4.0
MultiMate Advantage	2.0
Navy DIF	
Nota Bene	3.0
Novell Perfect Works	2.0
Office Writer	4.0 - 6.0

Format	Version (if applicable)
OpenOffice Writer	1.1 - 3.0
Oracle Open Office Writer	3.x
PC File Doc	5.0
PFS:Write	Versions A, B
Professional Write (DOS)	1.0, 2.0
Professional Write Plus (Windows)	1.0
Q&A Write (Windows)	2.0, 3.0
Samna Word IV	1.0 - 3.0
Smna Work IV+	
Samsung JungUm Global (File ID only)	
Signature	1.0
SmartWare II WP	1.02
Sprint	1.0
StarOffice Writer	5.2 - 9.0
Total Word	1.2
Wang PC (IWP)	Versions through 2.6
WordMarc Composer	
WordMarc Composer+	
WordMarc Word Processor	
WordPerfect for DOS	4.2
WordPerfect for Macintosh	1.02 - 3.1
WordPerfect for Windows	5.1 - X4
WordStar 2000 for DOS	1.0 - 3.0
WordStar 2000 for DOS	2.0, 3.0
WordStar for DOS	3.0 - 7.0

Format	Version (if applicable)
WordStar for Windows	1.0
XyWrite	Through III+

Appendix B

Record Properties Generated by Crawling

During a crawl, the IAS Server produces record properties according to certain naming schemes. You can map any of these properties to Endeca attributes.

Common record properties

The IAS Server generates certain properties whether you crawl a file system, a CMS repository, or a custom data source.

The IAS Server generates record properties and assigns each property a qualified name, with a period (.) to separate qualifier terms. The IAS Server constructs the qualified name as follows:

- The first term is always `Endeca` and is followed by one or more additional terms.
- The second term describes a property category, for example: `CMS` or `FileSystem`. The term `File` may be added to files from either file system or content management repositories.
- The third and fourth terms, if present, fully qualify the property, for example: `Endeca.CMS.ItemId` or `Endeca.FileSystem.Path`.

The IAS Server may generate the following properties for all records:

Endeca Property Name	Property Value
`Endeca.Action`	The action that was taken with the document. Values are `UPSERT` (the file or folder has been added or modified) or `DELETE` (the document or directory has been deleted since the last crawl).
`Endeca.SourceType`	Indicates the source type of the crawl. Values are `FILESYSTEM` (for file system crawls), `WEB` (for Web servers), `CMS` (for CMS crawls), or `EXTENSION` (for custom data source extensions).

Endeca Property Name	Property Value
`Endeca.Id`	Provides a unique identifier for each record. For file system crawls, `Endeca.Id` is the same as `Endeca.FileSystem.Path`. It is the full path to the file including the file name. For archive files, this is a string pointing to a file within `Endeca.FileSystem.Path` a container. This property also includes the `PathWithinSourceArchive` (if present). For Web crawls, `Endeca.Id` is the same as `Endeca.Web.Url`. For CMS crawls, `Endeca.Id` is the concatenation of the `Endeca.CMS.RepositoryId` and `Endeca.CMS.ItemId` properties, and the `Endeca.CMS.ContentPieceId` (if present). This property also includes the `PathWithinSourceArchive` (if present) For custom data source extensions, a plug-in developer must add `Endeca.Id` to each record and assign it a value appropriate for the data source.
`Endeca.SourceId`	Indicates the name of the data source. This is the same as the `id` value of `crawlId` in a crawl configuration.
`Endeca.File.IsArchive`	A Boolean that, if set to a value of `true`, indicates that the document is an archive file, such as a Zip file. If the file is not identified as an archive, the property is absent. Note that archives are identified by their file extension or Mime type. It is possible for a document to have both `Endeca.File.IsArchive` and `Endeca.File.IsInArchive` properties set, as archive files may contain other archive files nested within.
`Endeca.File.IsInArchive`	A Boolean that, if set to a value of `true`, indicates that the document is extracted from an archive file. If the file is not an archived document, the property is absent.
`Endeca.File.Size`	The size of the document in bytes, as reported by the native file system, CMS, or an archive entry.
`Endeca.File.SourceArchiveId`	This property is added to all records that have the `Endeca.File.IsInArchive` property. It is intended to provide a reference to the original archive that was encountered in the file system or CMS. The value is the original archive's `Endeca.FileSystem.Path` or `Endeca.Id` property. In the case of nested archives, it is the top-level archive, because that is the original source in the file system or CMS being crawled.

Endeca Property Name	Property Value
`Endeca.File.PathWithinSourceArchive`	This property is added to all records that have the `Endeca.File.SourceArchiveId` property. The value of this property is the path to the current record within the source archive file. In the case of nested archive entries, it includes the path to the nested archive, appended with the path to the current record within the nested archive.

Record properties generated by file system crawls

During a file system crawl, the IAS Server produces record properties according to a standardized naming scheme.

Windows file example

The following example shows the properties returned from a Windows crawl for a Windows text file named `TestFile.txt`, which is owned by user fsmith from the DEVGROUP domain:

```
...

<RECORD>

...

  <PROP NAME="Endeca.FileSystem.Owner">
      <PVAL>DEVGROUP\fsmith</PVAL>
  </PROP>
  <PROP NAME="Endeca.FileSystem.Group">
      <PVAL>DEVGROUP\Domain Users</PVAL>
  </PROP>
  <PROP NAME="Endeca.FileSystem.IsHidden">
      <PVAL>false</PVAL>
  </PROP>
  <PROP NAME="Endeca.FileSystem.IsTemporary">
      <PVAL>false</PVAL>
  </PROP>
  <PROP NAME="Endeca.FileSystem.Path">
      <PVAL>c:\endecafiles\TestFile.txt</PVAL>
  </PROP>
  <PROP NAME="Endeca.FileSystem.ParentPath">
      <PVAL>c:\endecafiles</PVAL>
  </PROP>
  <PROP NAME="Endeca.FileSystem.ACL.AllowRead">
      <PVAL>BUILTIN\Administrators</PVAL>
  </PROP>
  <PROP NAME="Endeca.FileSystem.ACL.AllowRead">
      <PVAL>NT AUTHORITY\SYSTEM</PVAL>
  </PROP>
  <PROP NAME="Endeca.FileSystem.ACL.AllowRead">
      <PVAL>DEVGROUP\fsmith</PVAL>
  </PROP>
  <PROP NAME="Endeca.FileSystem.ACL.AllowRead">
      <PVAL>BUILTIN\Users</PVAL>
  </PROP>
  <PROP NAME="Endeca.FileSystem.IsDirectory">
      <PVAL>false</PVAL>
  </PROP>
```

```
  <PROP NAME="Endeca.FileSystem.ModificationDate">
      <PVAL>1182453853873</PVAL>
  </PROP>
  <PROP NAME="Endeca.FileSystem.CreationDate">
      <PVAL>1182453827530</PVAL>
  </PROP>
  <PROP NAME="Endeca.Action">
      <PVAL>UPSERT</PVAL>
  </PROP>
  <PROP NAME="Endeca.FileSystem.IsSystem">
      <PVAL>false</PVAL>
  </PROP>
  <PROP NAME="Endeca.File.Size">
      <PVAL>16</PVAL>
  </PROP>
  <PROP NAME="Endeca.Document.Type">
      <PVAL>Unknown (ASCII 8)</PVAL>
  </PROP>
  <PROP NAME="Endeca.Document.Text">
      <PVAL>This is a test.</PVAL>
  </PROP>
</RECORD>

...
```

Common file system properties

The IAS Server produces some common properties from records crawled in either a Windows or UNIX file system.

The following record properties are common to documents fetched from both Windows and UNIX file systems.

Endeca Property Name	Property Value
`Endeca.FileSystem.Extension`	The file extension of the document, which is the string after the last dot in the file name. If the document has no dot in the name, this property will not be generated.
`Endeca.FileSystem.Group`	The name of a group for which permissions have been set for the document. For UNIX files, the property value is the `groupname`. For Windows files, the name is prepended with the domain to which the group name belongs, in the format: `DOMAIN\principal`.
`Endeca.FileSystem.IsDirectory`	A Boolean that indicates whether the document is a directory (a value of `true`) or a file (a value of `false`). The value is set to `true` even if the directory is in a container.
`Endeca.FileSystem.IsHidden`	A Boolean that indicates whether the document is a hidden file (a value of `true`) or not (a value of `false`).

Endeca Property Name	Property Value
`Endeca.FileSystem.ModificationDate`	The date when the file was last modified. Modifications include changing permissions on the document. The date format is in milliseconds since midnight January 1, 1970 UTC (Coordinated Universal Time).
`Endeca.FileSystem.Name`	The name of the file.
`Endeca.FileSystem.Owner`	The name of a user or other principal who is the owner of the file. For UNIX files, the property value is the `ownername`. For Windows files, the name is prepended with the domain to which the name belongs, in the format: `DOMAIN\principal`.
`Endeca.FileSystem.Path`	The identifier of the full path to the file, including the file name. For archive files, this is a string pointing to a file within a container. This property also includes the `PathWithinSourceArchive` (if present).
`Endeca.FileSystem.ParentPath`	The identifier of the path to the directory containing the file. This does not include the file name. For archive files, this is a string pointing to a container.

Record properties for file system crawls on Windows

The IAS Server produces certain properties from records crawled on a Windows file system.

The following table lists the record file properties that are specific to Windows file systems.

Endeca Property Name	Property Value
`Endeca.FileSystem.ACL.AllowRead`	The name of a user, group, or other principal who has the right to read the document. The name is prepended with the domain to which the name belongs, in the format: `DOMAIN\principal`.
`Endeca.FileSystem.ACL.DenyRead`	The name of a user, group, or other principal who is denied the right to read the document. The name is prepended with the domain to which the name belongs, in the format: `DOMAIN\principal`.

Endeca Property Name	Property Value
`Endeca.FileSystem.CreationDate`	The date when the document was created. The date format is in milliseconds since midnight January 1, 1970 UTC (Coordinated Universal Time).
`Endeca.FileSystem.IsSystem`	A Boolean that indicates whether the document is a system file (a value of `true`) or not (a value of `false`).
`Endeca.FileSystem.IsTemporary`	A Boolean that indicates whether the document is a temporary file (a value of `true`) or not (a value of `false`).

Record properties for file system crawls on UNIX

The IAS Server produces certain properties from records crawled in a UNIX file system.

The following table lists the record file properties that are specific to UNIX file systems.

Endeca Property Name	Property Value
`Endeca.FileSystem.IsGroupReadable`	A Boolean that indicates whether the group (the `Endeca.FileSystem.Group` value) has read rights to the document.
`Endeca.FileSystem.IsOwnerReadable`	A Boolean that indicates whether the file owner (the `Endeca.FileSystem.Owner` value) has read rights to the document.
`Endeca.FileSystem.IsSymbolicLink`	A Boolean that indicates whether the document is a symbolic link that refers to another file or directory (a value of `true` indicates that the document is a symbolic link).
`Endeca.FileSystem.IsWorldReadable`	A Boolean that indicates whether everyone on the system (`world`) has read rights to the document.
`Endeca.FileSystem.LinkTarget`	The name of the document to which a symbolic link refers. This property is present only if the `Endeca.FileSystem.IsSymbolicLink` property is set to `true`.

Limitations with ACL properties

The Integrator Acquisition System on Windows cannot get ACL properties for seeds that represent a root folder. However, the Integrator Acquisition System successfully gets ACL properties for all children of the root.

This limitation only occurs in the following scenario:

- A file system crawl has the `gatherNativeFileProperties` option enabled.
- The machine running the Endeca IAS Service is a Windows machine.
- The seed specified represents a root folder (for example, `C:\` or `\\machinename\folder`).

The Integrator Acquisition System produces an Endeca record for a root folder, and the record is tagged with other generated record properties. Only the ACL properties are missing.

Document Conversion properties

The IAS Document Conversion Module generates certain properties for records crawled with document conversion enabled.

The IAS Document Conversion Module generates `Document` properties that contain information (including the text) of the document or metadata about the document.

Endeca Property Name	Property Value
`Endeca.Document.Metadata.`*`attribute`*	Metadata information in the document. The metadata attributes depend on which ones were added by the authoring tool used to create the document. For example, an Adobe Acrobat PDF document could have such metadata attributes as `Endeca.Document.Metadata.title` and `Endeca.Document.Metadata.primary_author`.
`Endeca.Document.Metadata.Misc`	Properties that are returned from the IAS Document Conversion Module but that do not have a type attribute are mapped to this property.
`Endeca.Document.Text`	The text (content) of the source document. Note that the IAS Document Conversion Module typically does not preserve line break information.
`Endeca.Document.TextExtraction.Error`	An error returned by the IAS Document Conversion Module. Note that a `no filter available for this file type` error indicates that you should modify the document conversion module to exclude files of this type.

Endeca Property Name	Property Value
`Endeca.Document.Type`	The type of document, such as Microsoft Word 2003/2004, Adobe Acrobat (PDF), JPEG File Interchange, and Extensible Markup Language (XML).

Note:

- You should not use these properties for filters. These properties are created after the files are accessed, and therefore cannot be used to filter out files.
- If you crawl a data source without text conversion enabled, none of these properties are generated.

Record properties generated by CMS crawls

The IAS Server produces certain CMS properties regardless of whether document conversion is enabled or not. The following record properties are common to CMS crawls.

Endeca Property Name	Property Value
`Endeca.CMS.Uri`	The URI of the object which, if defined, allows an application to access an object as a web resource.
`Endeca.CMS.UpdatedBy`	The user name of the person who updated the content item.
`Endeca.CMS.RepositoryType`	The type of CMS repository, such as `Documentum Content Server`.
`Endeca.CMS.RepositoryVersion`	The version of the CMS repository, such as `7.1`.
`Endeca.CMS.RepositoryId`	The ID of the repository, such as `MySharepoint`.
`Endeca.CMS.ItemId`	The unique ID of the item in the repository.
`Endeca.CMS.ContentPieceId`	The unique ID of the item's content piece.

Endeca Property Name	Property Value
`Endeca.CMS.Path`	The path to the item in the repository, including the name of item, such as `/dctm65/test`. In cases where an item resides in several locations, the property value for both `Endeca.CMS.Path` and `Endeca.CMS.ParentPath` depends on which location the IAS Server encounters first when crawling the repository. This means that subsequent crawls of the repository may produce different property values. For example, if an item named `doc1` resides in `root/folder1/doc1` and `root/doc1`, the IAS Server can produce a property value of either `root/folder1/doc1` or `root/doc1`.
`Endeca.CMS.ParentPath`	The path to the parent of the item in the repository, not including the name of the item, such as `/dctm65`.
`Endeca.CMS.Name`	The name of the item.
`Endeca.CMS.Author`	The author of the item.
`Endeca.CMS.IsFolder`	`true` if the item is a folder, `false` otherwise.
`Endeca.CMS.NumContentPieces`	The number of pieces of content associated with the item in the repository.
`Endeca.CMS.ContentLength`	The length in bytes of the content as reported by the IAS Server.
`Endeca.CMS.CreationDate`	The creation date of the item.
`Endeca.CMS.ModificationDate`	The last modified date of the item.
`Endeca.CMS.MimeType`	The MIME type of the item.
`Endeca.CMS.AllowReadContent`	An ACL entry for a user or group that can read the content of the item.
`Endeca.CMS.DenyReadContent`	An ACL entry for a user or group that cannot read the content of the item.
`Endeca.CMS.AllowReadProperties`	An ACL entry for a user or group that can read the properties of the item.
`Endeca.CMS.DenyReadProperties`	An ACL entry for a user or group that cannot read the properties of the item.

Here are additional notes concerning record properties produced by CMS crawls:

- In addition to the properties listed above, an Endeca record may also contain properties that are specific to a CMS repository that are passed through to IAS. Such properties have a prefix of `Endeca.CMS.Misc`.

- CMS crawls may be inconsistent in creating the format of ACL property values. For example, property values could contain: user display name, user display name@domain, domain\user name, domain\group name, or domain\role name.

  IAS returns names in the form [domain\](user name), [domain\](group name), and [domain\](role name), but IAS is limited by the capabilities of the underlying CMS and the values the CMS returns in ACLs.

How CMS crawls handle multiple pieces of content

Some CMS repositories support items with multiple pieces of content. In these cases the IAS Server outputs a record for the item and records for each piece of content.

For example, an item from the Documentum Content Server repository could contain an attached PDF and an Excel file.

After the crawl, the records for each piece of content will contain:

- All properties of the original item record, such as ACL user and group permission entries of type `Endeca.CMS.AllowReadContent`
- A content piece identifier property `Endeca.CMS.ContentPieceId`
- An identifier of a specific record `Endeca.Id`. It is the concatenation of the `Endeca.CMS.RepositoryId` and `Endeca.CMS.ItemId` properties, and also the `Endeca.CMS.ContentPieceId` (if present).

Example of generated records for items with multiple pieces of content

This example includes a portion of output for two records — the first is the root document that has two pieces of attached content. The second is the first of the attached pieces. The Id property is produced by concatenating the `RepositoryId` with the `ItemId`, plus the child record's `ContentPieceId` (if present), using a colon as a delimiter (shown in bold in the example):

```
<?xml version="1.0"
encoding="UTF-8"?>
<RECORDS>
 <RECORD>
    <PROP NAME="Endeca.Action">
       <PVAL>UPSERT</PVAL>
    </PROP>
    <PROP NAME="Endeca.CMS.ContentLength">
       <PVAL>0</PVAL>
    </PROP>

    ...

    <PROP NAME="Endeca.CMS.Name">
       <PVAL>doc_with_attachment</PVAL>
    </PROP>
    <PROP NAME="Endeca.CMS.NumContentPieces">
       <PVAL>2</PVAL>
    </PROP>
    <PROP NAME="Endeca.CMS.RepositoryId">
       <PVAL>discussion</PVAL>
    </PROP>

    ...

                <PROP NAME="Endeca.Id">
       <PVAL>discussion:doc_with_attachment</PVAL>
    </PROP>
```

```
    <PROP NAME="Endeca.CMS.ItemId">
       <PVAL>doc_with_attachment</PVAL>
    </PROP>
    <PROP NAME="Endeca.CMS.RepositoryType">
       <PVAL>Documentum Content Server</PVAL>
    </PROP>
    <PROP NAME="Endeca.CMS.RepositoryVersion">
       <PVAL>release 6.5</PVAL>
    </PROP>
    <PROP NAME="Endeca.SourceType">
       <PVAL>CMS</PVAL>
    </PROP>
    <PROP NAME="Endeca.SourceId">
      <PVAL>DocumentumSource</PVAL>
    </PROP>
 </RECORD>
 <RECORD>
    <PROP NAME="Endeca.Action">
       <PVAL>UPSERT</PVAL>
    </PROP>
    <PROP NAME="Endeca.CMS.ContentLength">
       <PVAL>54699</PVAL>
    </PROP>
    <PROP NAME="Endeca.CMS.ContentPieceId">
       <PVAL>Attached.pdf</PVAL>
    </PROP>
    <PROP NAME="Endeca.CMS.RepositoryId">
       <PVAL>discussion</PVAL>
    </PROP>

    ...

    <PROP NAME="Endeca.Id">
       <PVAL>discussion:doc_with_attachment:attached.pdf</PVAL>
    </PROP>

    <PROP NAME="Endeca.CMS.IsFolder">
       <PVAL>false</PVAL>
    </PROP>
    <PROP NAME="Endeca.CMS.ItemId">
       <PVAL>doc_with_attachment</PVAL>
    </PROP>
    <PROP NAME="Endeca.CMS.MimeType">
       <PVAL>application/pdf</PVAL>
    </PROP>

    ...

    <PROP NAME="Endeca.Document.Type">
       <PVAL>Adobe Acrobat (PDF)</PVAL>
    </PROP>
    <PROP NAME="Endeca.File.Size">
       <PVAL>54699</PVAL>
    </PROP>
    <PROP NAME="Endeca.SourceType">
       <PVAL>CMS</PVAL>
    </PROP>
    <PROP NAME="Endeca.SourceId">
       <PVAL>DocumentumSource</PVAL>
    </PROP>
 </RECORD>

...

</RECORDS>
```

Index

R

S

T

U

W

Z