Load Data and Content

Overview

The Attivio Intelligence Engine (AIE) can ingest both structured data and unstructured content (referred together as just "content") from databases, email systems, file systems and more.

See Content Ingestion - Concepts and Tools, which names and diagrams all of the parts that make ingestion work.

This is the parent page over many child pages that describe AIE's connectors.

This page continues with links to pages that describe various parts of the ingestion process in more detail.

> If AIE is running in a **low-memory (less than 8GB)** environment, see the Memory Usage Tuning guide before feeding large volumes of content into AIE.

View incoming links. **>>**

Activating Incremental Updating , Architecture , Attivio Glossary , Attivio Platform , Configure the Attivio Schema , Connectors , Content Ingestion - Concepts and Tools , Creating Custom Scanners , In Depth Architecture Diagrams , Index Engines , Message Ordering

Loading Content

The process of gathering content and processing it in AIE is referred to as **ingestion**.

Content can be loaded through three mechanisms:

- Client APIs
- AIE Connectors

- Command-line utility: **aie-exec** controlConnector

Each connector delivers content to a workflow that is configured for that type of content. This allows for an almost limitless array of processing options.

Most of the connector topics (child pages to this one) show how to configure the connector using an editor in the AIE Administrator.

Examples of using the Client APIs to load content can be found in the following guides:

Main article: Java Client API

Connector Components

A typical connector consists of a scanner, a message publisher, and a result listener.

Scanners

A scanner is the underlying Java class that acquires content for a connector. (The terms "scanner" and "connector" are sometimes used interchangeably in AIE. Strictly speaking, a scanner is one part of a connector.)

Scanners implement the logic necessary to convert the source data format into an AttivioDocument.

One can create custom scanners using the Java Server API.

Main Article: Creating Custom Scanners

Message Publishers

A Message Publisher (sometimes called a "feeder") is the part of a connector that takes AttivioDocuments from the scanner and sends them to an ingestion workflow. It is common to use the default publisher, which is the **DirectMessagePublisher.**

Schema Configuration

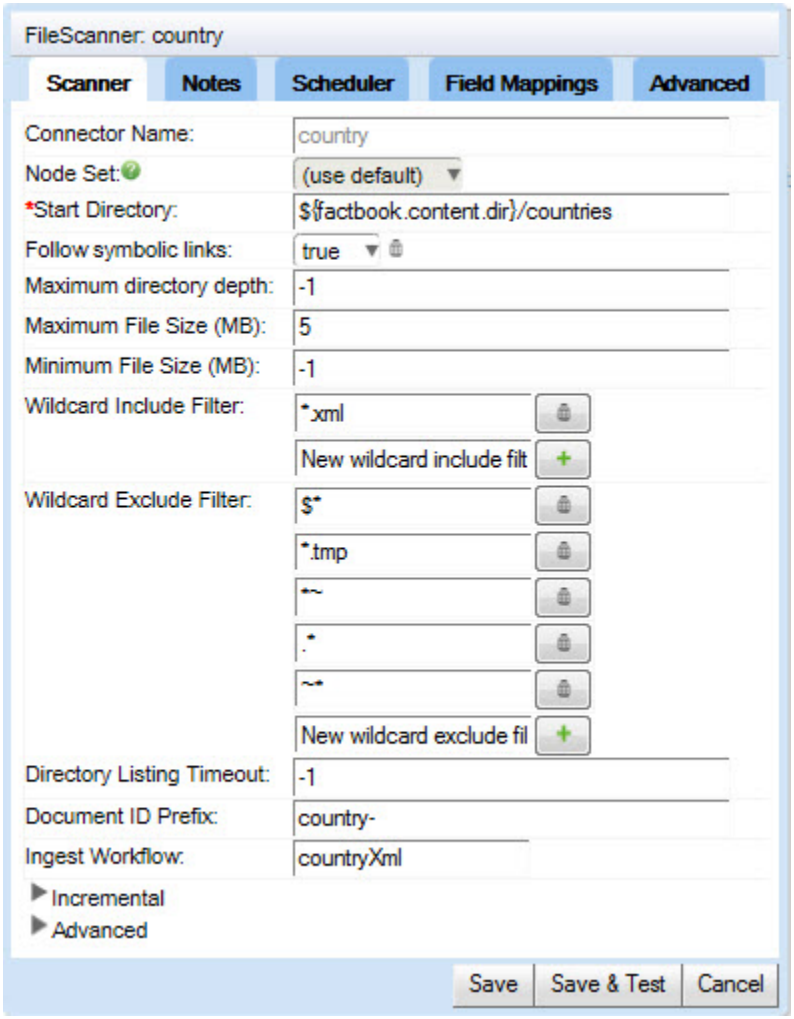
AIE Schema as a defines what AttivioDocument fields should be stored and indexed. If the field is not defined in the schema, the index engine will ignore it. The schema must be tailored to fit your incoming documents and records.

AIE Schemas require far less time to set up than traditional schemas, as AIE provides dynamic field definitions and does not require relationships between fields to be defined prior to ingestion.

Main article: Configure the Attivio Schema

Connector Configuration

Configuring a connector is very simple using the New Connector tool in the AIE Administation Web Interface. This tool lets you select a connector type from a list, and then opens an editor with default parameters in place for all available fields. This is the country connector from the Factbook demo.



To complete the configuration, simply give the connector a name, supply the location of the content, and enter the name of the target workflow.

Main article: Connectors

Processing Content

AIE provides several mechanisms for processing and enriching content during the ingestion process.

Text Extraction

AIE extracts text and metadata from files in a wide range of formats.

Main article: Advanced Text Extraction Module

Linguistics Processing

Linguistic processing makes incoming text indexable, and lays the groundwork for many forms of enrichment.

Main article: Linguistic Analysis

Text Analytics

Text analytics extracts interesting terms and phrases from unstructured text.

Main article: Text Analytics.

Deleting Content

AIE provides several mechanisms for deleting documents from the index.

Main article: Deleting Content

Updating Content

Updating AIE content is the same as adding new content with the exception of Real-time Field Updates.

Main article: Updating Content

Incremental Updates

AIE can track which data or content has changed and only process new, modified and deleted items.

Main Article: Activating Incremental Updating
Tutorial Example: Incremental Updating Example

Connector Locations

Connectors can be specified to run on a nodeset in the topology.

Main Article: Multi-Node Topologies

Connector Lifecycle

To simplify object lifecycle management and minimize configuration complexity, scanners, message publishers and result listeners are all created every time the connector is run. If the connector is asked to begin a new crawl while it is currently running, the request is ignored and the current crawl is completed. Once the connector crawl is complete, all connector objects are destroyed.

Remote Connectors

It is possible to create a Java application that feeds AttivioDocument messages to a remote instance of AIE.

Main Article: Ingest Application Example.

Main Article: Monitoring External Connectors.

Configuring File Based Connectors For HDFS

Connectors which use file based scanners such as the 'Generic File System' scanner, the 'XML files' scanner and the 'CSV files' scanner can ingest data from HDFS (Hadoop File System) as well as from Linux and Windows FS. The following steps must be performed in order to configure AIE to ingest data from HDFS:

- Stop all AIE processes excluding the agent.
- Follow the instructions in Set Up Zookeeper to configure AIE to access the Hadoop cluster.
- Restart the AIE processes.

When the file based connector is configured, set the 'File System URI' field to HDFS:// - any URI info following the HDFS:// string will be ignored since AIE uses the information configured in the Set Up Zookeeper step above to access the Hadoop cluster.

If HDFS is secured by Kerberos, a principal and a keytab file must be configured as well.

- Update the project's attivio.core-app.properties file like the following:

```
#Principal/Keytab for kerberos authentication
security.hadoop.principal=<principal name>
security.hadoop.keytab=<path to keytab file>
```

The above principal/keytab pair is used as the default HDFS access credentials. Alternative principal/keytab pair can be configured for each connector under the Scanner-->Kerberos tab.