Overview

This project is an MVP / PoC enabling indexing of documents (in common formats like PDF, Word / Excel / PowerPoint, TXT) stored in **ThingWorx** repositories with the help of **Elasticsearch** and then performing full text searches against the documents' content from **ThingWorx** (mashups or services).
The project is a simple and small and only consists of 4 **ThingWorx** entities, 2 of which are there only for demo purposes.
The implemented functionality only covers the main use case - performing full-text search. Advanced stuff, flexibility and additional features are to be implemented depending on the requirements of the particular project and intended use cases.
See the accompanying video for the installation steps and the demo / quick overview.

Prerequisites

ThingWorx 8.5+
Elasticsearch 7+ with the ingest attachment plugin installed. Pipeline name should be "attachment", like in the example provided in the plugin documentation.

Installation of **Elasticsearch** is out of scope of this document, but it can be done really quickly by following the official installation guide

Installation

As the result of the installation, the following four entities will be created in **ThingWorx**:

1. The key entity - Thing: ElasticHelper.xml
   It has 4 services:
   - *bulkLoadElastic* - bulk loads / indexes documents from repository to **Elasticsearch**.
   - *deleteElasticIndex* - a small service which just does exactly what one would expect
   - *genSHA1* - generates SHA1, which is used as _id for **Elasticsearch** documents
   - *queryElastic* - queries data from **Elasticsearch**, used in mashup. Can be used standalone.
2. Data Shape: ElasticResult_DS.xml
   Used to store output from the *queryElastic* service to display data in mashups
3. Mashup: Elastic.xml
   Just a sample mashup for demo / showcasing purposes.
4. Repository Thing: ElasticSample_Repository.xml
   An auxiliary thing for storing sample data for the test / demo purposes.

Installation steps:

1. Import entities into **ThingWorx** (Import -> From File)
2. Copy the "ElasticSample_Repository" folder with the sample data to the *\ThingWorxStorage\repository* folder
3. Set appropriate values for the properties of the *ElasticHelper* Thing

Property	Default value	Comment

base64Prop		Was used for debugging purposes
elasticIndexName	**ThingWorx**	Index in **Elasticsearch** used for storing data
elasticserver	http://localhost:9200	Address of **Elasticsearch** server
repository	DOC_Repository	Name of **ThingWorx** repository to index
thingworxserver	https://tw9.irisoft.ru:8443	Address of **ThingWorx** server

Validating the installation

Make sure that all the properties on the *ElasticHelper* are set to correct values and **Elasticsearch** instance is up and running.
Execute the *bulkLoadElastic* service on the *ElasticHelper* Thing with *debug* and *testCase* parameters set to True.

> Setting *testCase* to True forces the app to use ElasticSample_Repository repository in **Thingworx** and test index in **Elasticsearch**
> Files from the *ElasticSample_Repository* will be indexed and *test* index in **Elasticsearch** will be created.
> The output of the service is in json format and should resemble the example below.

```
{
    "total": 4,
    "indexed": 4,
    "doc": {
        "/archives/Dump1.txt": "updated",
        "/ErrorLog.log": "updated",
        "/ConfigurationLog.txt": "updated",
        "/archives/Log1.txt": "updated"
    }
}
```

If the service returns something like the above, it pretty much means that things were installed correctly.
You can set the value of the *elasticIndexName* property to "test" (no quotes) and open the *Elastic* mashup and enter a search term.
Remember to revert the *elasticIndexName* property back to the "correct" value

> You can also do a quick check that **Elasticsearch** index was created by visiting **Elasticsearch** _indices page (e.g. http://localhost:9201/_cat/indices) and/or doing a simple query like http://localhost:9200/test/_search?q=IIoT&pretty

Indexing a repository

Make sure that all the properties on the *ElasticHelper* are set to correct values, **Elasticsearch** instance is up and running.
Execute the *bulkLoadElastic* service on the *ElasticHelper* Thing with *debug* and *testCase* parameters set to False
Depending on the amount of documents in the repository, process can take a while, but normally it takes under a minute.
If the index is new, as soon as the first document is indexed, the index will be created and should be visible in the list of **Elasticsearch** indices (http://localhost:9201/_cat/indices)
Open the *Elastic* mashup and enter a term for searching. The results should appear in the Grid below the input field.

The query in the *queryElastic* service is just a basic "query_string" on the "attachment.content" field and one would probably want to modify it to allow for more advanced stuff like fuzzy searches, sorting, partial matching etc.

ToDo / not implemented

1. loading / indexing a single document
2. deleting a single document from **ThingWorx** and **Elasticsearch** collection
3. working with multiple indices and/or repositories.
   Probably should have a ThingTemplate (instead of current ThingHelper) and multiple Things for each repository, but the implementation would really depend on business use case.
   Currently possible workarounds are:
   - either indexing data from multiple repositories into the same **Elasticsearch** index (should works as it is).
     Setting the "repository" property is required at the moment of bulk loading data
   - or switching the "elasticIndexName" and "repository" properties "on the fly".
     A small extra service would probably be required