# Unified RESTful Search Service for Structured and Unstructured Documents

20-Jan-2014

This document describes at a high level a few items of interest at the beginning of the project.

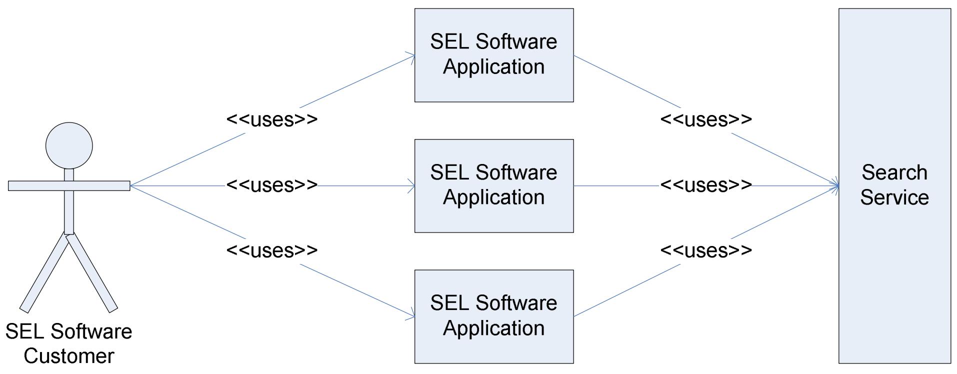

**Figure 1: Search Service Module Context View**

As shown in Figure 1, the Search Service's consumer is other software. Of course the SEL Software Customer stick man will benefit from the Search Service, but only through the lens of the other applications.

Given that, which admittedly doesn't convey much (more will come later), the challenge for you is at this stage of the project is multifaceted: You've got to learn about the problem space, figure how to build it and test a solution, and how to manage the parts and artifacts.

## How to manage the parts and artifacts

One thing you should determine early on is how and where your source code and documents will live. I highly recommend Git for version control, along with regular backups.

We are big fans of continuous integration. You should be too. I recommend Jenkins as a build server.

Also, for testing, what are the options for automated testing of RESTful service apis?

## How to build it and test a solution

Like most software projects, I expect this one to involve a mix of existing off the shelf software and new code. So, what kind of existing software is out there that can give us a hand?

We at least need a RESTful api server. What's out there? Keep in mind that the server is where much of your new code will live, so you'll want something flexible, or maybe just decide to roll your own.

We want the service to have a secure api. What are the options? What is the best practice?

Finally, both http://www.elasticsearch.org/ and http://lucene.apache.org/solr/ look very promising as search back-ends. Are there others?

## Learn about the problem space

There is a lot more to come in this area. For now there are a few things that need looking into.

First, we can assume some documents originate from the SEL Software Applications. How do those apps put documents “into” the search service. How can those apps restrict a search to just application specific documents and not the whole superset of searchable documents?

Also, we want the search results from structured documents include structure-specific metadata regarding the precise location of the search result within in the documents. For example, consider this semi-contrived XML document:

```
<RTACModule>
  <Device>
    <Name>SEL_Coupler_1</Name>
    <Connection>
      <Protocol>EtherCATClient</Protocol>
      <ConnectionType>Ethernet</ConnectionType>
      <SettingPages>
        <SettingPage>
          <Name>Properties</Name>
          <Row>
            <Setting enabled="false">
              <Column>Setting</Column>
              <Value>Vendor</Value>
            </Setting>
            <Setting enabled="false">
              <Column>Comment</Column>
              <Value>No comment</Value>
            </Setting>
          </Row>
        </SettingPage>
      </SettingPages>
       </Connection>
  </Device>
</RTACModule>
```

If our SEL Software Application in Figure 1 wants to find a setting value with “Ven”, we’d expect our search service to have a hit. How is this hit conveyed back to the calling application such it can act upon the result?

A related question for XML is how do we provide a schema/mask for XML documents – because not everything in an XML document is interesting to search.