One Search - A Simplified Search

Avvaru SAI CHANDU

Prahlad modi

## INDEX

[1. EXECUTIVE SUMMARY 2](#_Toc447296195)

[2. PROBLEM STATEMENT 3](#_Toc447296196)

[3. SOLUTION THROUGH CASE STUDY 4](#_Toc447296197)

[4. ARCHITECTURE 7](#_Toc447296198)

[5. BUSINESS BENEFITS 8](#_Toc447296199)

[6. CONCLUSION 9](#_Toc447296200)

[7. CONTACTS 10](#_Toc447296201)

[8. APPENDIX 11](#_Toc447296202)

## 1. Executive Summary

Majority of the enterprise applications consist of many business functional modules and individual search pages to search for specific entities within the business modules. As the size and complexity of the application increases, the number of search pages will increase at the rate proportional to the number of business modules in the application, resulting into poor user experience and low productivity of the end user.

This eminence paper proposes an approach and solution to resolve such issues and provides a detailed approach to handle multiple search pages in enterprise applications through a single search page.

“One Search” solution provides a unique approach to resolve search page issues that can be used in any enterprise applications having multiple search pages. With the help of “One Search”, the user can perform search on a single search page with the desired query to achieve the results without the tedious job of navigating through many levels of menu items or pages to achieve the same search results.

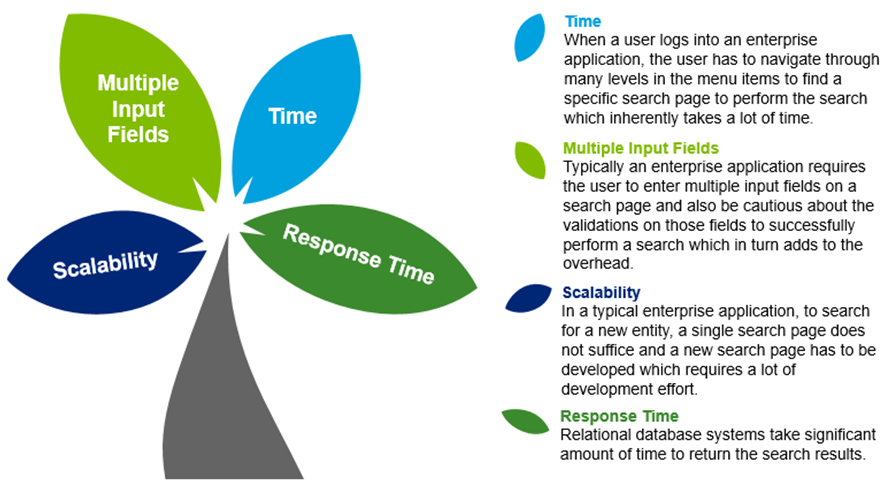
A case study on a State Public Sector system is performed to assess the proposed solution and the outcome. Case study application consists of more than 16 business modules having multiple search pages for each module. When an end user logs into this application system, the user has to perform many actions and has to navigate through many levels in the menu items to find a specific search page to perform the search.

A lack of technology is not the reason for the on-going dissatisfaction with enterprise search that exists within many organizations. It is about building and maintaining an enterprise search excellence application which is highly functional, scalable and reliable.

## 2. Problem Statement

Effective enterprise search represents one of the most challenging areas in business today. The whole area of search has been revolutionized by Online Search Engines like Google, Bing etc. Clients now expect to be able to locate relevant data and navigate as easily as they do on the web through Google and other search engines in their private lives. When this ease of search is not replicated in our professional lives it can be quite frustrating. As we create more content than ever before, the importance of effective search across the enterprise continues to grow.

Here are the problems identified in traditional enterprise search systems:

****

## 3. SOluTion Through Case Study

For large enterprise applications complexity poses a significant threat. With new addition of a new module the complexity of the application grows significantly. The main challenge and failure of large Enterprise applications is Complexity-Creep.

Complexity can be managed through forward planning and the application of best practices to design and implementation. One search is one of the solutions to avoid these kind of problems in large enterprise applications.

To access application data, One Search is a natural and powerful extension of the familiar, language-based, single search box and search results page user interface which allows more users to easily access and leverage information managed in enterprise applications. Users will benefit from increased productivity through reduced training, quicker intuitive access, and unified multi-source results.

A state eligibility determination application called “Worker Portal” has been considered for case study. The following section provides detailed information about the current implementation followed by an implementation with OneSearch and its advantages.

To access ‘Task Details’ screen in the current worker portal application, the following steps are to be followed:

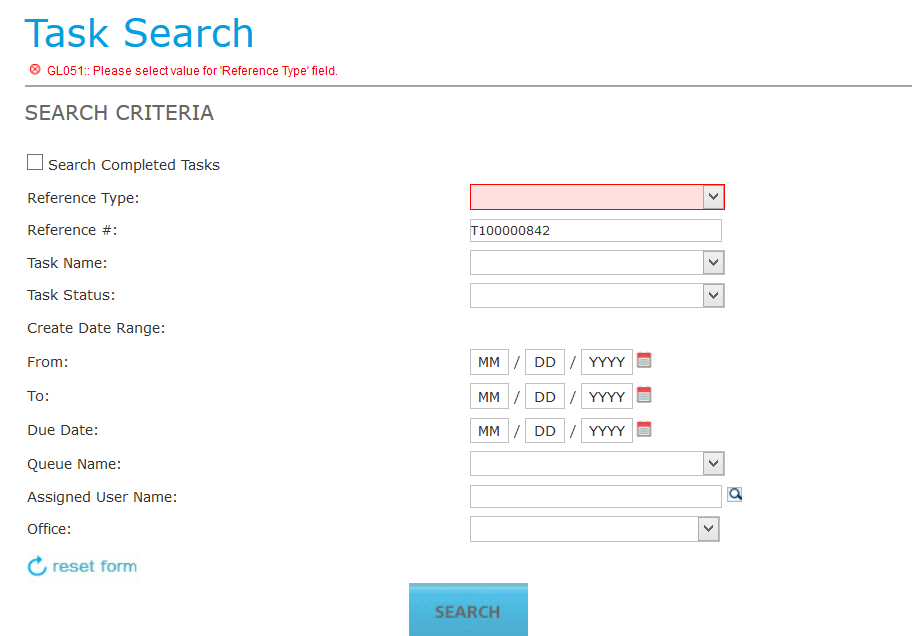
1. Login into the Application
2. Click on *“Others”* from the menu tab.
3. Click on “*Task Management”* from the Menu extension.
4. Click on “*Task Search”* from the sub-menu.
5. Enter the details in the respective fields based on the search criteria, as shown in Figure 3.1.
6. After retrieving the data, click on edit icon in the *“Search Results”*

**Impediments to search in the current “Task Search” screen**

* In the Search Screen, we have different conditional validations for different fields on search page. The reason for adding the validations on the search page is to minimize the overhead on the relational database and to narrow the search for the database.

For example:

*GL051: Please select value for 'Reference Type' field.*Validation message comes in the *“Task Search screen”* when user enters only the *Reference Id* without selecting the *Reference type***.** This is a typical case of conditional validation on the search screens in Worker Portal Application.

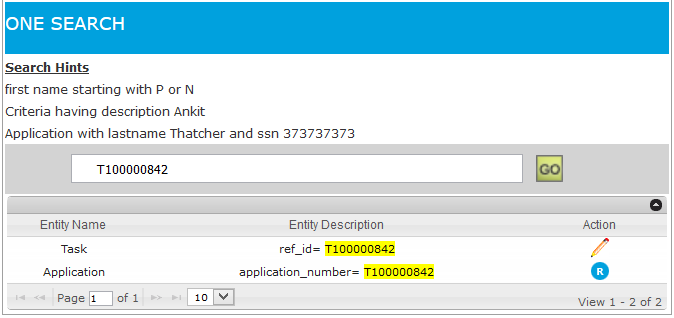


**Figure 3.1:** Task Search screen with multiple options for the user to search

* As there are validations on the page for *Search Criteria* and the user has to give all the data so that the all the conditions are met, thereby increasing the overhead for the user.

To access ‘Task Details’ screen using “One Search” solution in the Worker Portal Application, we have to follow the below steps:

1. Login into the Application.
2. Enter the Search Criteria in the Search box using Simple English like *‘Task with reference id T100000842’ (or)*directly*‘T100000842’*.
3. After retrieving the data, click on edit icon in the *“Search Results”*.



**Figure 3.2**: Task search screen with OneSearch

Following are the points which substantiate our claim that “One Search” leapfrogs traditional search system in a huge way

* The number of steps for reaching the required page is always 3 in “One Search” Application irrespective of the number of entities.
* The User saves the time and do not have the overhead to understand the validations specific for the entity search.
* New search page is not required as “One Search” handles multiple entities, so the user will not have worry about the navigation for the new search page in the Application. “One Search” needs only minor backend changes to implement the new Search Functionality.
* “One Search” uses *Indexed Columns* Solr DB for fetching the Results, so it is faster in fetching the results when we compare to the relational database which we generally use in our enterprise Applications. In relation databases, indexing all columns in a table might lead to increase in response time for CUD operations.
* Also, the navigation is not affected and the user can directly navigate to the desired results page.

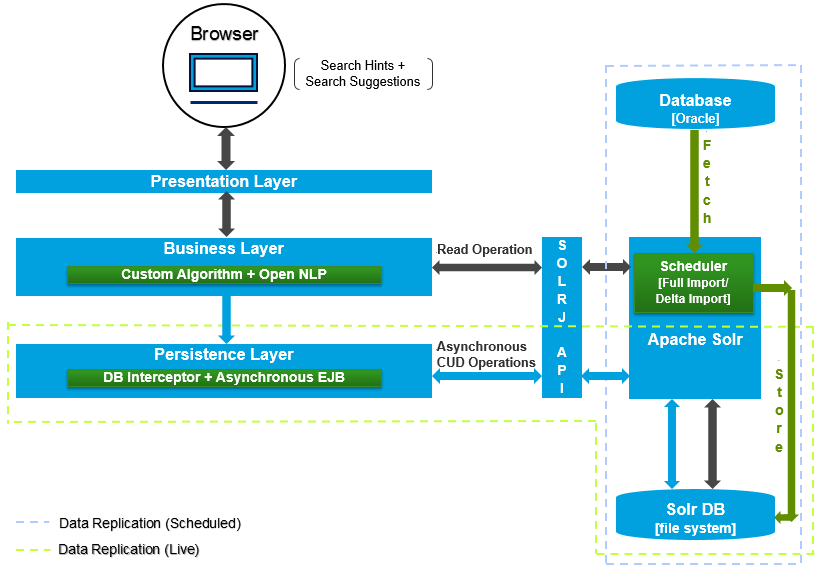
In this way, “One Search” improves the functionality and makes it user friendly without affecting the existing modules in the enterprise Application.

## 4. ARCHITECTURE

OneSearch application architecture is based on a standard 3-tier JEE application architecture consisting of presentation, business and persistence layers as shown in Figure 4.1. Detailed information about few important components of this application is described below. Detailed technical implementation of this application is provided in the appendix section.

**4.1 Search Suggestions Alogrithm:**

It suggests the suitable keywords/parameters of the system entities and helps user to quickly build the search query.

**4.2 Criteria Formation Algorithm:**

It analyzes and parses user queries with the help of NLP to create context based search. OpenNLP library is a machine learning based toolkit for the processing of natural language text. The two important techniques used are:  
1. *Parts-of-Speech Tagging*  
2. *Parsing*  
Along with these, the algorithm uses the *entity mappings* to process and build the search query. This query/criteria is sent to search engine through Java API - SOLRJ.

**Figure 4.1**: OneSearch Application Architecture

**4.3 Search Engine:**

Apache Solr is a standalone enterprise search engine comprising of Server, Rest API, Data Handlers and Data Analyzers. Solr internally uses Lucene to index and search the data. Solr receives and parses the criteria to provide search results and sends the response back through *http* in different data formats like JSON, XML, Text etc.

**4.4 Data Replication(Scheduled):**

A HttpPostScheduler is configured and plugged into the Solr exploded .WAR to enable the data replication from the Application Database to Solr Database. DataConfiguration property file which contains configuration like datasource, queries, fields against each entity is placed in Solr Core/Profile. This property file is used by the ‘HttpPostScheduler’ to replicate the data.

**4.5 Data Replication(Live):**

Live Data replication is achieved using a Database Interceptor. The Interceptor will make a call to Asynchronous process(Asynchronous EJB) to convert the Data Access Object to the Solr Document and will update the Solr Database through SOLRJ API.

## 5. BUSINESS BENEFITS

Using an enterprise search engine to access application data is a natural and powerful extension of the familiar, language-based, single search box and search results page user interface which allows more users to easily access and leverage information managed in enterprise applications. “One Search” envisions search as the primary point of integration and of access to all enterprise modules and data sources. Clients will benefit from increased productivity for employees, managers and executives through reduced training, quicker intuitive access, and unified multi-source results.

One Search is a product that enables you to find information within your enterprise application by keyword or contextual searches. One search can be extended to provide simple, fast and secure access to modules within a variety of Enterprise applications (e.g. DES, etc.), and to use this capability as a distinguishing user experience advantage.

* **User Experience:** The UX/UI has been significantly improved in a way that the user can perform the search using simple English.
* **Scalability:** Single search page with single search field can handle multiple entities search in the application. Searching a newly added Entity will be a breeze.
* **Time and Effort:** Traditional way of developing a search involves a Jsp, EJB (backend bean) and DAO call. The development/Enhancements can be speeded up with “OneSearch” Solution implementation. Also, the time taken to navigate to various search pages in the application is curtailed.
* **Response Time:** “OneSearch” uses indexed based search which generates results in very short span of a time compared to Oracle SQL search. To get a great search, database’s text search just won't do, it just doesn't have Solr's extensive feature-set nor under-the-hood access to tweak things from analysis to scoring.
* **Too many input fields:** “OneSearch” just has one text box for search and is independent of the entity the user is searching for thereby eliminating the need of multiple input fields.
* **Direct Navigation:** Users can go directly to an application menu function or option based on keywords. For example, typing in “Individual id: 132561253” should be able to provide a link that can take the user directly to an application with the requested individual id.
* **Visualization:** Finally, the visualization of information specific to a data source can further enhance the productivity of the end user. For example, instead of showing a standard hit list of all the search results, it might be more useful to show a simple table for each hit that contains the relevant information (e.g. entity name, description, navigation link etc.) upfront in an easy-to-understand format.

## 6. CONCLUSION

There is no doubt that the search landscape is changing towards a more natural and spontaneous language based search instead of traditional search page. OneSearch is the solution and approach that can be used in any Enterprise solutions, custom development projects, enterprise applications having multiple search pages to achieve better user experience and improved productivity while performing the search with the desired criteria on a single search page.

The solution is not in better technology, because the technology is already good enough for most purposes. It lies in applying proven processes, best practices, and appropriate expertise to support the technology.

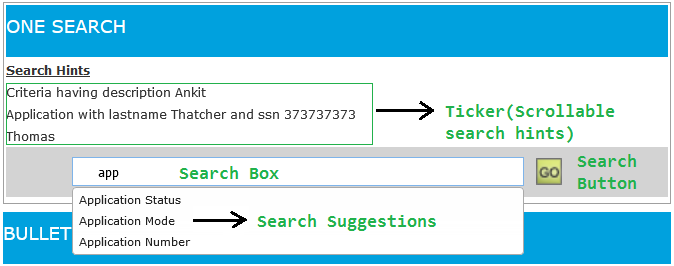
The strategic advantages of making better use of corporate assets are tough to illustrate on a simple spread sheet. The day-to-day marginal gains in business effectiveness, agility and productivity are tough to assign specifically to enterprise search…

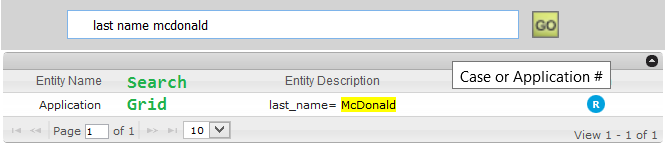
But they exist, and they make a difference.

## 8. Appendix

**8.1 COMPONENTS OF “OneSearch” PAGE**

* Ticker - The search page has **scrollable search hints** (Fig 8.1.1) which has been implemented using jQuery.
* Search Box - An autocomplete enabled text field (Fig 8.1.1) where in the user can input a search query.
* Search Suggestions - A search algorithm has been implemented to help the user with appropriate suggestions (Fig 8.1.1) to build a search query.
* Search Button – A button with the label ‘GO’ (Fig 8.1.1) performs an Ajax call to fetch the search results.
* Search Grid- A jQuery grid (Fig 8.1.2) to show the results with actions associated with the respective entities.

  
**Fig 8.1.1**

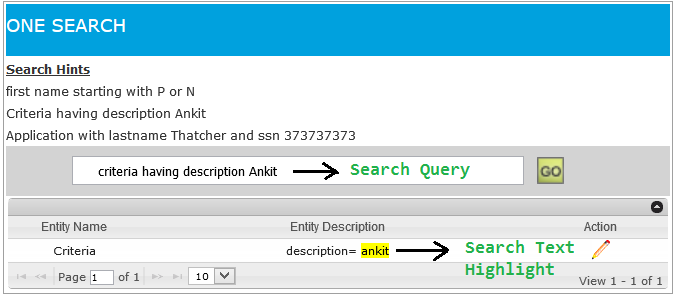
  
**Fig 8.1.2**

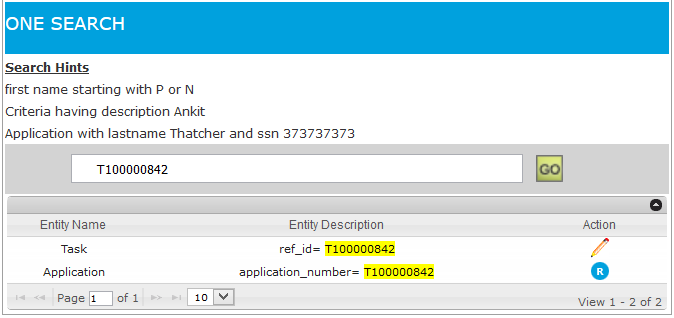
**8.2 BEHIND THE SCENES**

The AJAX call is fired once the user clicks on the ‘GO’ button. The search input query is analyzed and processed by the **Open NLP**. NLP processes the natural language using *‘Parts-of-speech’ and ‘Parser’* techniques and forms the tokens. These tokens are further analyzed by the **Criteria Formation Algorithm** using a **Mapping file** (entity-field-solrcolumn mappings) to create the search criteria.

The search criteria is subsequently sent to **SOLR** using **SOLRJ API**. **SOLR** will eventually process the request and responds back with relevant search results. The results are processed and actions (Edit/Read-only etc.) are associated against respective entities and the AJAX response is sent back. The results are then shown to the user in the Search Grid (Fig 8.1.2).

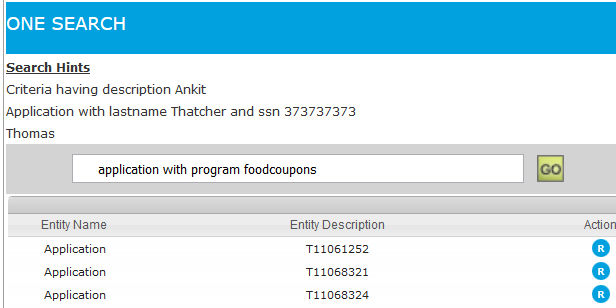
The searched values are highlighted (Fig 8.2.1) in the search results displayed in the **Search Grid**.­­ Search navigators (aka dynamic navigation) provide a reliable and effective general solution to the search results (Fig 8.2.2).

 **Fig 8.2.1**

 **Fig 8.2.2**

**8.3 Semantic Search**

The OneSearch also searches for the user Input based on the Semantics.

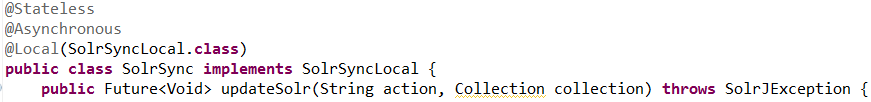
**  
Fig: 8.3.1**

**8.4 The Data**

The data to Solr Database is managed in two ways:

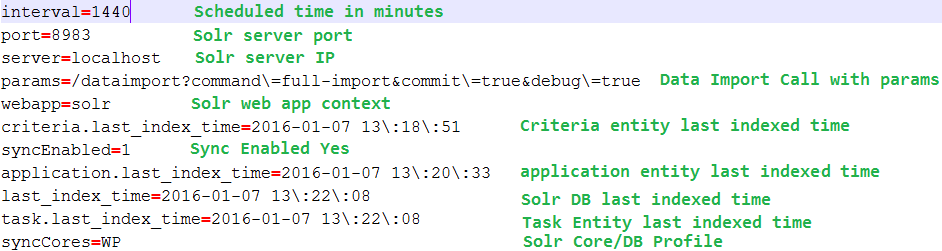
* Live
* Scheduled

Live: A Database interceptor is identified in the WorkerPortal i.e. FwDataDirector. An Asynchronous EJB is written to handle the calls to Solr asynchronously. The asynchronous method (Fig 8.4.1) is invoked **only after the application database call is successfully executed** in order to maintain the data consistency.

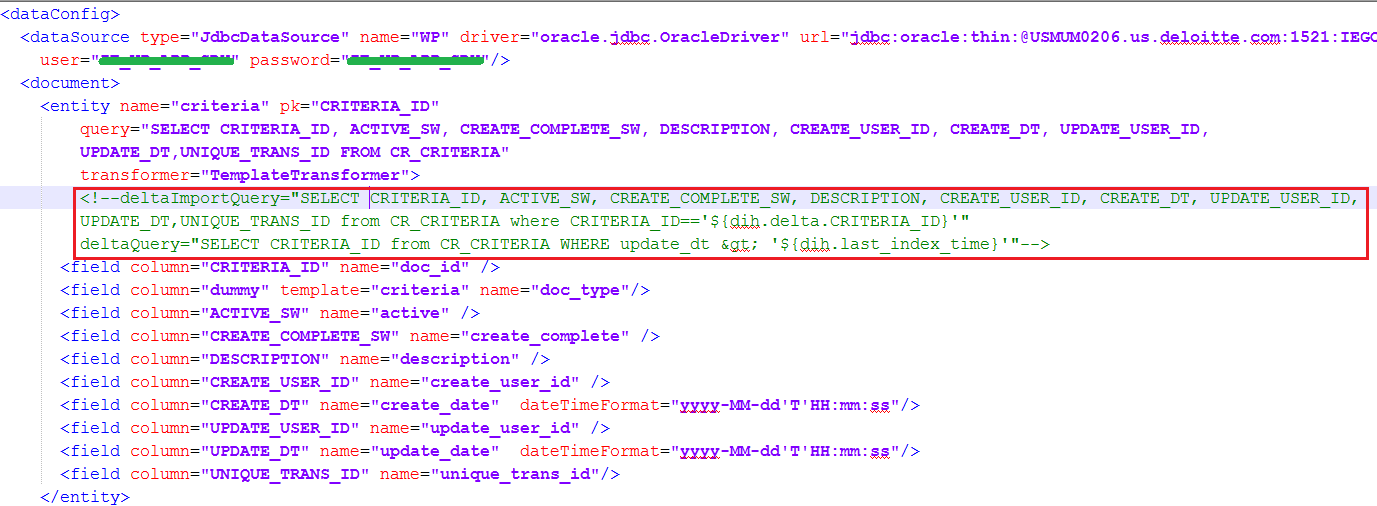


**Fig 8.4.1**

Scheduled: A **HttpPostScheduler** is configured to run after every 24 hours and plugged into the Solr exploded .WAR to facilitate the data replication from the **Application Database to Solr Database**. This scheduler uses *DataImport, DataConfig* property files to import the data.

* DataImport configuration property file (Fig 8.4.2) which contains properties of Solr server IP, Port, DataImportHandler Url, Solr Core etc. is placed in Solr Core/Profile.
* DataConfiguration property file (Fig 8.4.3) which contains configuration like datasource, queries, fields against each entity is placed in Solr Core/Profile.  
  

**Fig 8.4.2**

****

**Fig 8.4.3**

The scheduled Import can be configured in two ways.

1. **Full Import:** All the entities are selected for import from the application database.
2. **Delta Import:** A proper database designed table contains an additional column “last\_modified” of timestamp datatype. To download the table/entity again since it has been updated recently, we use this timestamp field i.e. *UPDATED\_DT* to determine what rows in each table have changed since the last indexed time. The *DeltaQuery* can be seen in Fig 8.4.3.