Distributed Code Search Engine for Enhancing Reusability in Enterprise Projects

Pragya Gupta
Computer Science & Engineering
Oriental Institute of Science & Technology Bhopal, Madhya Pradesh, India

Sreeja Nair
Computer Science & Engineering
Oriental Institute of Science & Technology Bhopal, Madhya Pradesh, India

Sridhar Jammalamadaka
Tata Technologies
Bangalore, Karnataka

ABSTRACT
In IT industry, one of the most important goal of the organization is to reduce overall cost of development. One way to achieve this is making the developers to write modular and reusable code organized in subroutines or packaged libraries. The difficult challenge is to search through code spanning across millions of lines used in various legacy projects. This paper explains about a framework that aims to develop a search engine which enables easy searching through large repositories of source code. It also goes further to achieve scalability and high performance through the deployment of search engine in a distributed system.

Keywords
Solr, Distributed Search, Zookeeper

1. INTRODUCTION
In IT industry common issue faced is the reinventing a code logic again and again. Large number of projects happens in an IT industry. A piece of code or logic that is used in one project can be used across other projects too. But this never happens because there is no common location or a standard method that lets the people search for the already existing codes that are generic. Customer proprietary codes can be excluded. Distributed code search engine is an application that helped in addressing this issue and enhancing reusability in enterprise projects. The main features of Distributed code search engine are user can reuse libraries and code snippets.

Solr is an open source enterprise search platform. It is built on Apache Lucene Project written in java. It uses the Lucene library for full- text search. It's includes faceted navigation, real time indexing, hit highlighting, rich document handling (e.g. Word,PDF), dynamic clustering ,NO SQL features and database integration. It's providing Distributed Search through Sharding and index replication. Solr is highly scalable and fault tolerance most popular enterprise search engine.

Solr is java application which runs as a standalone search server. Solr has REST (HTTP/XML and JSON API) that make it usable from other programming language. For full text indexing and search solr uses the Lucene java library at its core. Solr has plug-in architecture to support more innovative customization and it has powerful external configuration allows user to create many type of application without java coding.

2. DISTRIBUTED SEARCH
When the size of an index is too large and it is not fit on a single system or if a single query takes long time to execute, an index can be divided into many shards, and solr can query and merge results across those shards.

2.1 Solr cloud
Apache Solr includes the ability to set up a cluster of solr server. These clusters combine fault tolerance and high availability called solr cloud. Which provides distributed indexing and search capabilities with following features:-

1. Automatic load balancing and fail-over for queries
2. Zookeeper combination for cluster coordination and configuration
3. Central configuration for the entire cluster

Solr cloud is flexible distributed search and indexing. It has no master node to allocate nodes, shards and replicas. Depending on configuration files and schemas solr uses zookeeper to manage these locations. Files or documents can be sending to any server and zookeeper.For distributing indexed content and query request across multiple servers, solr cloud provides a highly available, fault tolerant environment. It's a system in which data is organized into multiple shards. These shards can be hosted in multiple machines with replicas, providing duplicate data for both scalability and fault tolerance and zookeeper server that manage the overall structure, so that routing of Indexing and Search request can be done properly.

2.1.1 Basic Concepts of Solrcloud
For a single node, Solr has a core that is basically a single index. If user wants multiple indexes, it is necessary to create multiple cores. But with Solrcloud, a single index can spread multiple Solr instances means a single index can be made up of multiple cores on different nodes. From the multiple cores one core that makes up one logical index is called a collection. For redundancy as well as index scaling, a collection can span in many cores and sole cloud has multiple collections. Collection can be divided into shards. Each shard can exist in multiple copies; these multiple copies of the same shard are called Replicas. Replica is a physical index. Within a shard one of the replicas is the Leader, which is deputing by a leader-election process.

2.2 Zookeeper
Zookeeper is a distributed, open-source service for distributed applications. It defines a simple set of primitives that distributed application can build upon to implement higher level services for integration, configuration maintenance, and groups and naming.

2.2.1 Design Goal of Zookeeper
Zookeeper is simple and allows distributed processes to organize with each other through a Znode (Shared hierarchical namespace consists of data register). Zookeeper data is kept in memory means it can achieve high throughput and low latency number.

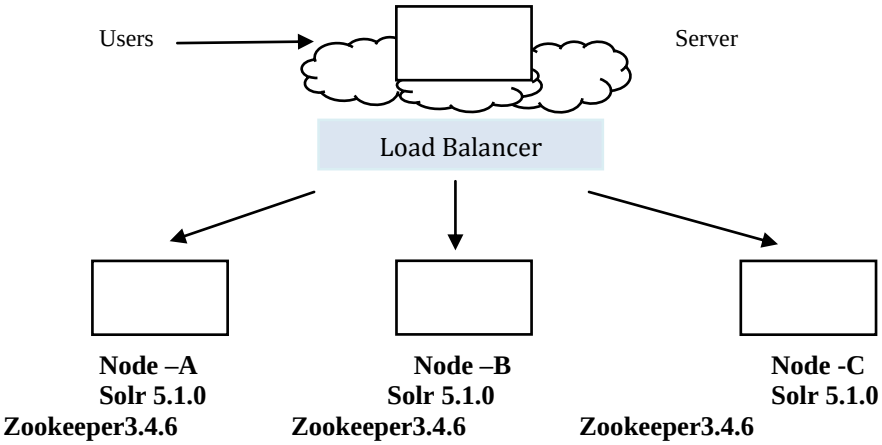


Fig 2.1: Architecture of Solr Cloud

3. WORK PLAN

Our proposed software uses three tier architecture,which is based on client server architecture. This architecture contains mainly two components first is solr search and second is Reusability software. In solr search, Lucene is a high-performance, full text search engine library written in java. In solr, the term core is used to refer to a single index and combination of transaction log and configuration files including schema.xml and solrconfig.xml among others. Solr installation can have multiple cores if needed, which allows to index data with special structures in the same server, and maintain more control over how data is presented to different audiences. Solr uses the Restful schema API to add fields to a managed schema. Consider a Web-based UI that offers Solr-as-a-Service where users need to organize a set of stop words and synonym mappings as part of an initial setup process for their search application. This use case can easily be supported by the Managed Stop Filter & Synonym Filter factories provided by Solr, via the Managed resources REST API.

The reusability software part contains three layers. The back end layer contains Database, and File Manager. Database is collection of an organized data. It is the set of schemes, tables, queries, reports, views and other objects .Database contains the relationship between data and information about the data so it can easily be accessed, managed, and updated. And file manager contains the code repository and project documents.

In Solr search the backend layer contains the Search Index. Solr is intelligent to get fast search responses because; it searches an index alternately, instead of searching the text directly. This is similar to retrieving pages in a book associated to a keyword by scanning the index at the back of a book, as different to incisive every word of each page of the book. This index is stored by Solr in a directory called index in the data directory.

The middle ware of reusability contains the POJO's, Jersey layer, Hibernate and Source Indexer. POJO is the basic old Java object, not bound by any special restriction. Jersey Restful Web Services framework is an open source, creation value framework for developing Restful Web Services in Java that provides support for JAX-RS APIs and serves as a JAX-RS (JSR 311 & JSR 339) orientation achievement. Hibernates Object –Relational mapping is a framework for mapping traditional relational database to an object oriented model in Java .

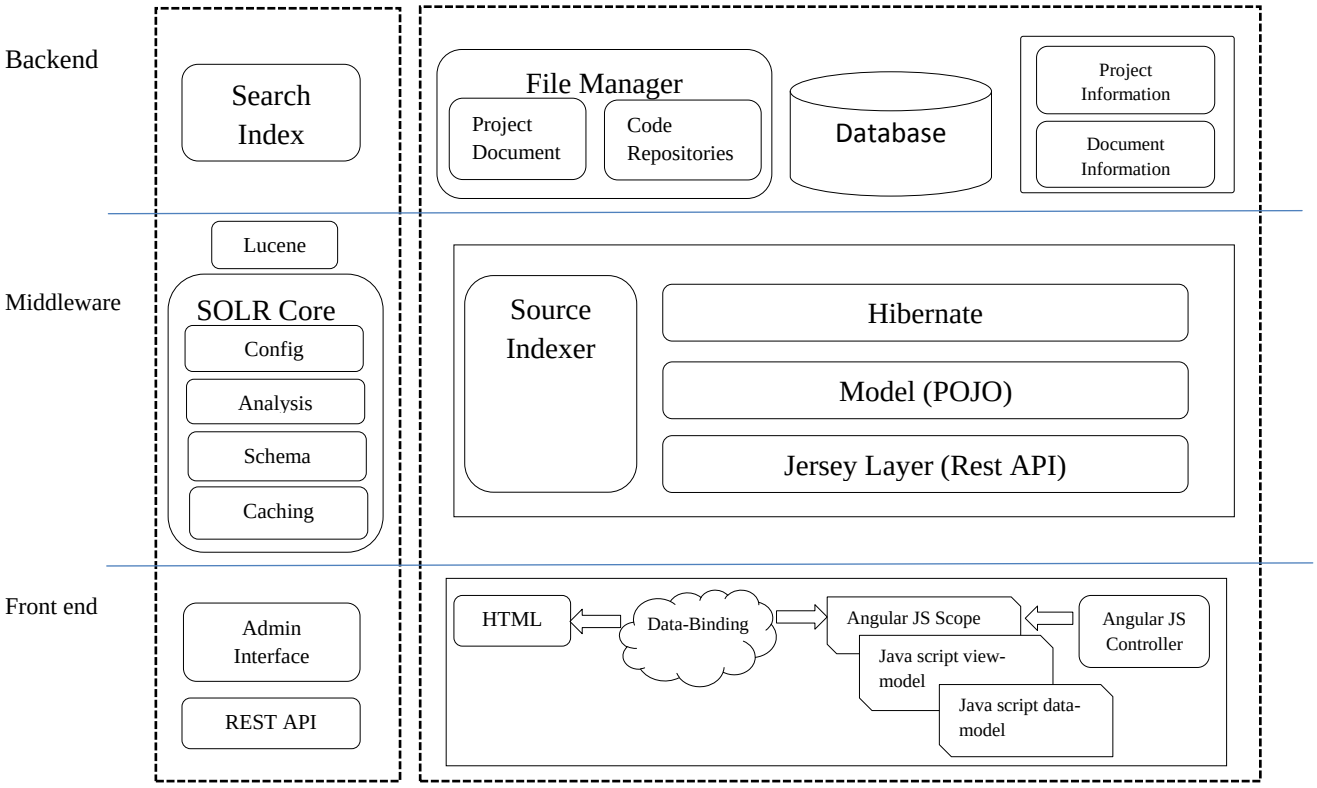


Fig 3.1: Architecture of Reusability

4. RESULT

1. 20M small Documents

   Elastic search indexed 20M documents in 900sec (approx.)

   Total Index size was 4 GB

   0.007 means, 99% < 0.03

   Solr indexed these documents in 1800 sec
   Total index size was 56 GB

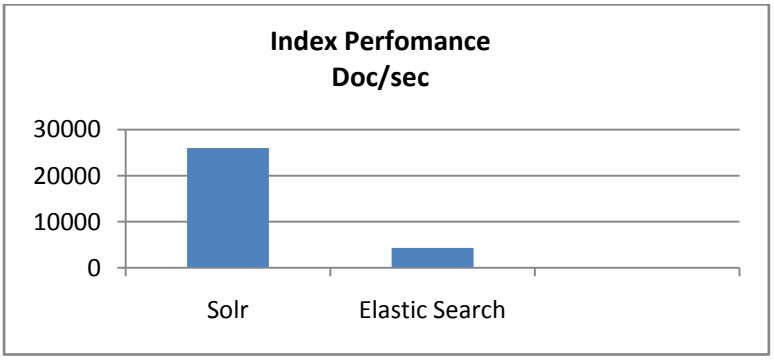
   Solr Indexed 20M documents in 1290sec

   Total Index size was 3.6 GB

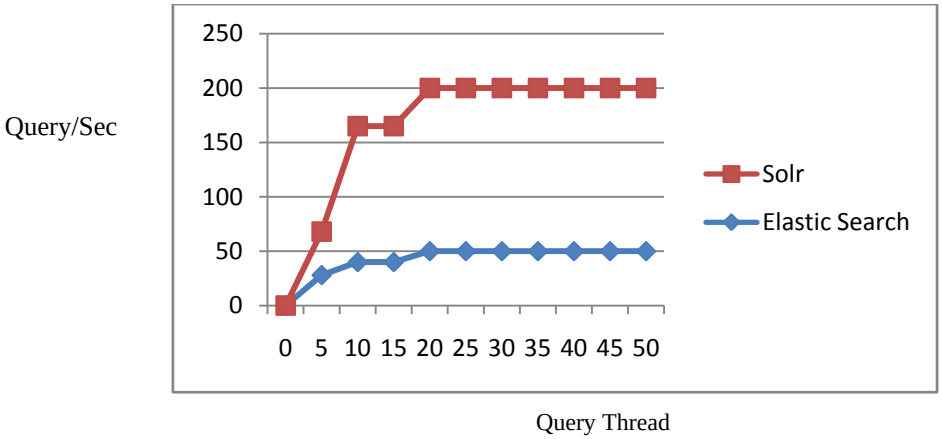
   0.007 means, 99 %< 0.05

2. 10M large Documents

   Elastic Search indexed these documents in 2700 sec
   Total index size was 91 GB



3. 20 M large documents
   Query per second



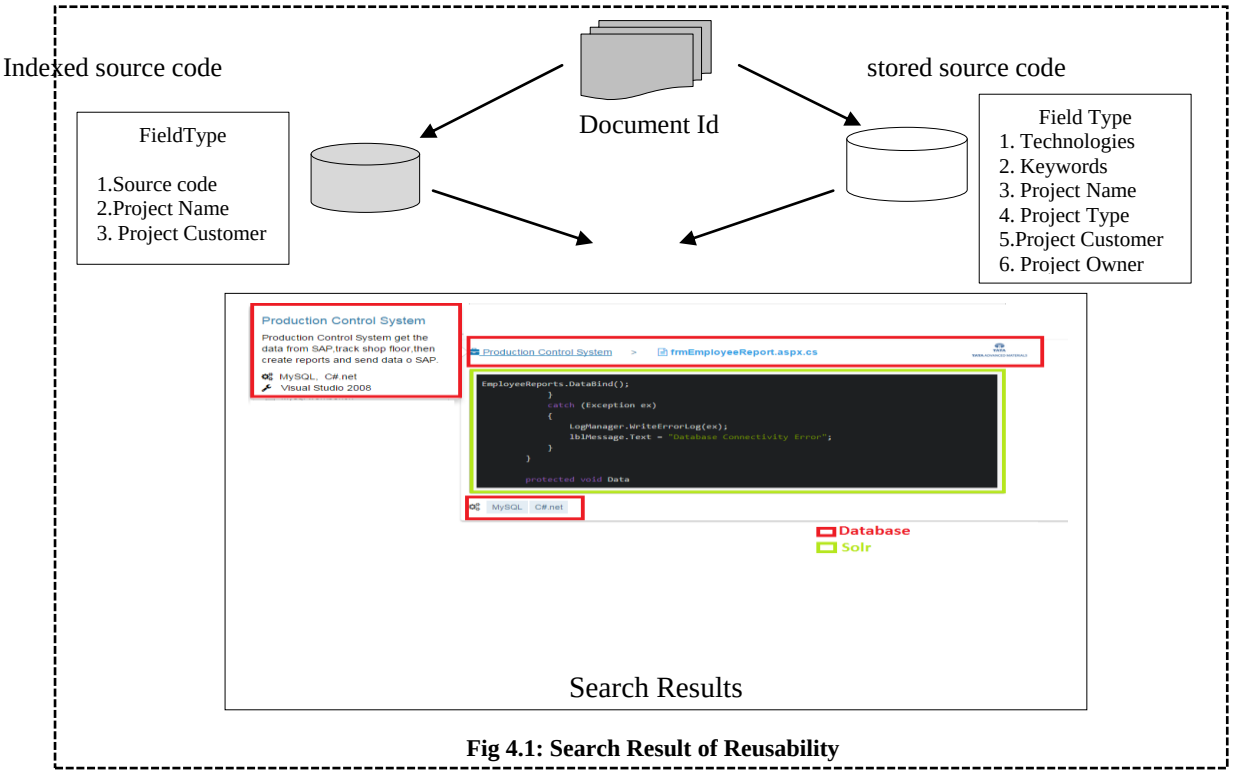


Fig 4.1: Search Result of Reusability

Solrcloud is comparatively faster than elastic search.Solrcloud can evidently support a significantly handle higher number of Query per Second indexing.

5. ADVANTAGES

This solution considerably reduces the development time and increases reusability. Since it is a web based application people can access this application with in the corporate domain in ease. It doesn't require any additional software setup on user machine. Solr Cloud Support Distributed indexing and Search capabilities. It's also Support Automatic load balancing and fail-over for queries. This application is reduce the development cost. This application has Scalable process and it's easy to deploy across the organization.

6. CONCLUSION

All the source code files are indexed by solr at a regular interval of time. When any new document is added the indexing can be done manually. The capabilities of Solr are made available to user by a web application front end. The data communication between the front end and Solr using rest layer. We used the following technologies to achieve seamless integration between the different elements. The user can search for a particular code using a search option that is available in the web application using keywords. There is an admin module which is used to add new source files to be indexed. The source codes are also classified based on the project name, language and domain. Search filters can also be obtained based on this category. To monitor the usage of this application and to provide security we have integrated the corporate login with this application. Search module contains Pagination, Facets, Code highlighter, Sort Filter, Search box, Document Detail, Download code.

7. FUTURE WORK

This search engine is Streamlining to development processes. But in future we can improve the search efficiency and easy way to add the new code. And also improve the presentation of search result. A custom plug-in can be created which can be integrated with the IDE used by the organization which eliminates the use of an extra application to access the reusable codes.

8. REFERENCES

[1] Mining modern repositories with elasticsearchKononenko, O Baysal, R Holmes

[2] http://lucene.apache.org/solr/ available at: APACHE SOLR

[3] https://zookeeper.apache.org/ available at: Apache ZooKeeper

[4] https://cwiki.apache.org/confluence/display/solr/SolrCloud

[5] https://www.elastic.co/products/elasticsearch

[6] https://searchcode.com/