ISI International Studies in Informatics
HAGENBERG

fh OBERÖSTERREICH UNIVERSITY OF APPLIED SCIENCES

JOHANNES KEPLER UNIVERSITY LINZ

softwarepark hagenberg

Name
Rami Henein
(Egypt)

Advisors
Dr.
Tudor Jebelean
Jakob Praher

Company
MIND BREEZE
Mindbreeze Software GmbH

Integrating Subversion Information with Source Code Analysis in Mindbreeze Enterprise Search Application

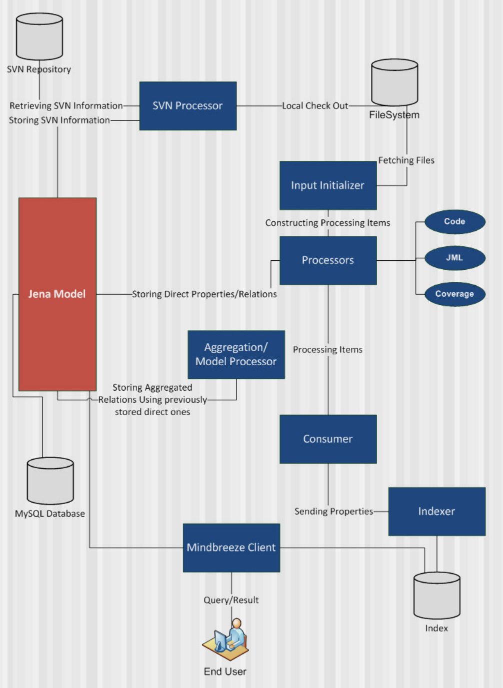
The continuous increase in the amount of information, exchanged between a variety of users and within a wide range of applications, has raised the need for more powerful and sophisticated search engines.
The approach presented in this thesis combines the normal source code parsing and analysis with other sources of information, namely Subversion repositories. Aiming for the realization of the suggested idea, a framework for source code search was built and integrated in Mindbreeze Enterprise Search System. Mindbreeze GmbH provides a powerful enterprise search system which combines the eficiency of search and the richness of information. The implemented framework provides a set of processors which extract several types of information from Subversion code repositories and source code files. The system is fully integrated with Mindbreeze Enterprise Search system.

Mindbreeze
Search
Subversion
Code Items
Source Code
Relations
Aggregation
Commit

Intro

In a world where information is a key factor and a crucial asset, the value of and the need for search engines is exponentially increasing with time. The development of application-specific search engines is one of the continuously emerging directions in the software world.
Inspiring its value from the ever growing field of software development, source code search is a recently tackled and evolving application. It offers new approaches to exploit existing source code for a variety of needs and for several types of users.

The majority of the attempts to develop source code engines was concerned with parsing the code and extracting the different programming language constructs in the form of abstract syntax trees. The work presented in this thesis has started with such an attempt to build a complete source code search and analysis system. The scope of the project was parsing and analyzing Java code and storing the results to be available for search. The project was integrated with an enterprise search system produced by Mindbreeze.
The developed system is limited by a single information source which is the abstract syntax tree resulting from the parsing of source code. Moreover, the data model representing the code elements was designed focusing on Java elements which is missing a required abstraction level in the design of such systems.



In order to have a more sophisticated system, a new approach was suggested for Minbreeze source code application. The main idea is to combine the information available from normal parsing with other types of information. Social information, formal code specifications and test coverage information are examples of other information sources which can be used to enrich the system and empower the search. Subversion, an automated version control system, is chosen as the source of social information as it provides useful metadata for source files as their log of changes or different modifiers. Moreover, it provides through its structure a detailed overview about software development projects by showing the stable and non-stable parts. For formal specifications, the Java Modelling Language was chosen as it is an evolving way to formally specify Java code.

In this thesis, combining source code parsing and analysis with social information source, namely Subversion repositories, is discussed in details. A framework is implemented as a base for the realization of the desired system. It should provide a flexible and configurable interface which allows the addition of several source code processors for extracting different types of information. The data model representing the di®erent elements has to be as generic as possible keeping the design simplicity. A convenient storage methodology is investigated and implemented. A set of concrete use cases is defined and realized within the framework implementation. Finally, the system is fully integrated within Mindbreeze Enterprise Search system.

Commit 2292 200930241
Kind: commit
Commit Number: 2292
Commit Log: new test created
Commit Author: eszter.tasi
Commit Date: Mon Jun 22 13:10:03 CEST 2009
Items Changed: XMLConstants.java , com.mindbreeze.codesearch.crawler.processor.concreteprocessors.JavaCodeProcessor, addAll, com.mindbreeze.codesearch.crawler.processor.concreteprocessors.CtagsProcessor, V5.IntStack.alma, pop, endElement, get, addAll, JavaDocsProcessor.java, C.java, size, IntStackInterface.java, TestRun, com.mindbreeze.codesearch.crawler.processor.ContentProcessor, toProcess, com.mindbreeze.codesearch.crawler.processor.concreteprocessors.JavaDocsProcessor, com.mindbreeze.codesearch.crawler.processor.concreteprocessors.XMLConstants, TestRun.java, getAppropiateField, com.mindbreeze.codesearch.crawler.processor.NWayMergedProcessor, toProcess, SequentialProcessor, merge, SequentialProcessor.java, b.c.OImpl, process, isEmpty, V5.IntStack, getAppropiateMethod, toProcess, indexOf, lastIndexOf, SpecificationXMLParser.java, OImpl.java, a.BInterface, generateKey, com.mindbreeze.codesearch.crawler.processor.concreteprocessors.CoverageXMLParser, getActualSrcFile, NWayMergedProcessor, OInterf.java, ContentProcessor.java, containsAll, parse, JavaCodeProcessor, com.mindbreeze.codesearch.crawler.processor.concreteprocessors.SpecificationXMLParser, BInterface.java, main, endElement, getAppropiateMethod, startElement, remove, process, getAppropiateClass, characters, V5.IntStackInterface, isempty, semmi, subList, process, retainAll, addCoverageInformation, ModelProcessor.java, main, listIterator, process, a.OInterf, IntStack.java, contains, run, clear, IntStackModel.java, com.mindbreeze.codesearch.crawler.processor.TestRun, remove, push, listIterator, add, set, addBasicInformation, process, addProcessor, add, CoverageXMLParser.java, com.mindbreeze.codesearch.crawler.processor.SequentialProcessor, ProcessorCollection.java, visitSourceFiles, com.mindbreeze.codesearch.crawler.processor.ModelProcessor, b.c.C, getProcesors, com.mindbreeze.codesearch.crawler.processor.AggregationAlgorithm, pop, CtagsProcessor.java, removeProcessor, startElement, iterator, toArray, V5.IntStackModel, a.AInterf, toArray, AggregationAlgorithm.java, removeAll, com.mindbreeze.codesearch.crawler.processor.ProcessorCollection, getAppropiateClass, resize, JavaDocsProcessor, addAlgorithm, AImpl.java, executeAggregations, NWayMergedProcessor.java, process, IntStack, AInterf.java, JavaCodeProcessor.java, process, Main.java, V5.Main, b.c.AImpl