GITHUB APPLICATION PROGRAMME INTERFACE AND WORDNET FOR CODE REUSE

Pirapuraj and Indika Perera

Department of Computer Science and Engineering, University of Moratuwa
Moratuwa, Srilanka
Pirapu @cse.mrt.ac.lk, indika@.mrt.ac.lk

Abstract

It is clear that code reuse is important task in software development and maintenance. As a lot of software application and source code have been used as libraries in version control systems, such that Git, SVN, LibreSource and related web sites, such that GitHub.com, sourceforge.net, projectsgeek.com, Googlecode.com, more and more companies, especially Small and Medium Enterprises (SMEs), are reusing open source code to develop their own software. The problem in code reuse is, after download all relevant code, we need to identify most relevant code among pool of code. In this paper we use keyword search with n-gram NLP technique using GitHub Application Program Interface (API). Before search the source code, we retrieve all Repository name in GitHub belongs to particular programing language (JAVA, C++, etc.), as well as we retrieve all .java file name if we search java libraries using GitHub API. Then compare our keyword with this list, if the keyword extracted from Software architecture is connected word, then we will split using Apache Camel Splitter. If the particular keyword related to any project, we download the project. Otherwise using WordNet, get some synonym and do the above process again. For further relevancy, we will use a speech recognition technique (Dynamic Time Warping (DTW)) and a NLP technique (Part of Speech Tagging (POS)). Because of this is a part of the whole research, in this paper we will consider only GitHub API.

Keywords: n-gram, GitHub API, WordNet, Dynamic Time Warping, Part of Speech Tagging, Apache Camel Splitte.

Introduction

When software architecture comes to development process developers may have to select appropriate libraries (as well as some sample code from other open source project in the Internet). This selection can be done either from their library repository of the own products or, by searching in the Internet for relevant available libraries. If the software company has any expert in specific area, they will suggest relevant libraries depending on their knowledge. However, to decide such manual processes are not feasible and inconvenient at a given time period and it is not proper way for big project. For an example if a software architecture includes 10 different classes, the developers may have to search 10 times for relevant libraries. So if they have a complex project in hand, it will be a challenge to them as well as it will take huge amount of time and cost. If there is any automated tool to do theseprocesses automatically, the time and the cost of the project would be considerably saved.

Open source code available on the Internet has become a common platform for sharing source code. Programmers often reuse the design of code examples or adapt code examples of existing open source projects rather than discovering usage patterns by digging into documents. Currently, the amount of open source code available on the Internet is enormous. For example, *sourceforge.net*1, the world's most popular website

for open source software development, hosts about 179,518 projects with two million registered users and a large number of anonymous users.

With such enormous amount of open source code available on the Internet, several code search engines (CSE) such as Google code search , Krugle , Koders , Sourcerer , and Codase are developed to efficiently search for relevant code examples (i.e., source files containing a search term). These CSEs accept queries such as the names of classes or methods of Application Programming Interfaces (API) and search in CVS or SVN repositories of available open source projects. [2]

Experienced workers are very important to companies, in the sense that someone with experience has higher productivity. When an experienced engineer or other qualified employee leaves a company, it takes with her/him a corpus of knowledge, which from the company point of view is a knowledge loss. In a high competitive market environment, where companies strive to be alive, this problem is crucial [19].

Research in software engineering has shown that software reuse positively affects the competitiveness of an organization: the productivity of the development team is increased, the time-to-market is reduced, and the overall quality of the resulting software is improved. Today's code repositories on the Internet provide a large number of reusable software libraries with a variety of functionality. The study results indicate that third-party code reuse plays a central role in modern software development and that reuse of software libraries is the predominant form of reuse. It shows that many of today's software systems consist to a considerable fraction of code reused from software libraries [1]. Therefore, code reuse is very important task in software development.

Code Searching

An eclipse plugin (CodeGenie) [15], which is based on Test-driven Code Search (TDCS) method for source code searching and reusing from open source project in the internet. In test-driven approach we need to write a test class before to all. The class includes input and expected output, then run the class, if it gets expected output success, otherwise failure. They used this approach to code search, but it has some drawbacks, before implement the actual program, the test class will be written, sometimes test program will be bigger than expected program. As well as guessing expected program is difficult. These are the drawbacks of this approach. But our keyword search with NLP technique is better way than this approach.

Steven P. Reiss et al. [3] showed that when follow keyword search, the first challenge finding appropriate keywords is difficult and another big problem is Keyword searches typically yield lots of unevaluated results. The problem with this is that programmers have to read each instance of returned code, attempt to understand what it does, and then determine if it meets their requirements.To make effective search semantics-based search is better way, the programmer need to state some information such that, what they want the identified code to do functionally, where it has to fit in, and etc. The proposed a tool (S^6) which allow to provide information about search. But, because of our proposed technique their concept is meaningless. We will easily find keyword from software architecture and for result evaluation we will use NLP techniques.

A prototype named as JBender [4], which increase the relevance of code search result with trust ability information. With keyword search, the developer cannot get a trustable project

from open source repository among a huge amount of open source project. So to get trustable project they use trust ability metrics for code search results that uses collaborative filtering of both user votes and cross-project activity of developers.JBendercreates a searchable index over the code base and provides acode search over it. Its novelty however lies in the underlyingmetadata which is linked to the projects in the searchablecode base upon finding results from the latter JBendercansupply the meta-information stored for the result's originatingproject.To collect meta-information they used Ohlohproject which is a social networking platform for open source software projects where projects (or rather their developers) can specify additional information. Ohloh is not a code search engine.

Ohloh provides user contributed information on both open source projects and their developers. Normally Ohloh gives Description of original project, project homepage, rating of the project, list of current repositories (type, url, last time of update), licenses of files in the project (exact type of license, number of files), employed programming languages (percentage of total, lines of code, comment ratio), the project's users and developers who worked on the project (experience, commits per project).Then they use those information they calculate trustability metrics. A project which gets higher trustability metrics value is a more trustable project. But using this approach, we can get information about trustable of a project, besides identifying of most relevant project is more difficult. As well as we are going to relate our search with the information extracted from software architecture, they did not include that.

We have implemented a framework (part of proposed tool for MSc) for helping reuse Java code. To implement this tool we integrated GitHub API to increase relevancy of code as well as it will help to check whether the GitHub repository contains the related project or not using the information extracted from Software Architecture. And downloads the code example results to form a local source code repository. Most of the Downloaded projects will be related to the information extracted from Software Architecture. But for further relevancy we need to do some NLP technique and speech recognition technique in future work because this is partial work of whole work.

This paper makes the following main contributions.

- An approach for reducing programmers' effort while reusing existing frameworks or libraries by providing relevant code samples.
- A program for collecting information from Software Architecture.
- A technique for collecting all java repository name and class name dynamically from GitHub using GitHub API.
- A technique for download related project from GitHub.

The rest of the paper is organized as follows. Section 2 explains our approach. Section 3 explain implementation details. Section 4 discuss limitation and future work, finally section 5 concludes.

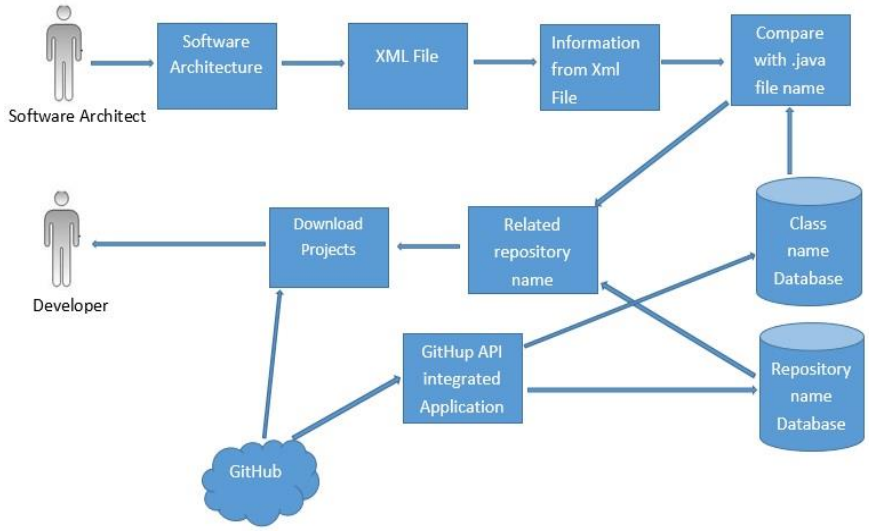

Figure 1: Overview of our Approach

Our approach consists of four major components, Extracting information from XML, comparison between Informationextracted from XML and .java files name, Getting related repository name, and GitHub API integrated application.

Figure 1 shows an overview of all components. Our approach address the described problem by accepting a software architecture (class diagram) in XML format, Then Extract some information from the software architecture (class name, methods name, attributes name). In this research we will focus on identifiers, program identifiers are a fundamental source of information to understand a software system. Because programmers choose program names to express the concepts of the domain of their software.

Extracting information from XML

Software Architecture can be converted to XML. In our approach, we will start from a Software Architecture (Class diagram) in XML format. Normally a class diagram consists of all detail about classes, methods, attributes and relations. Using aXMLparser, we will extract some information from the XML file. These information will be important to our approach, as well as it will be identifiers. So when we import a Software Architecture, automatically our approach will extract those information.

Comparison between Information and .java files name

Using GitHub API integrated application, we will get all .java file name list in GitHub. All identifiers extracted from XML file will be taken one by one and it will be compared with all .java file name list local database. If identifiers match with any .java, the particular project will be downloaded. Otherwise using WordNet (a lexical database of English word and sense relations. A *sense* is a particular meaning of a word. For each sense of a particular word, WordNet provides the *synset*, a list of synonyms for that sense.) All synonyms to particular identifier will be retrieved and again repeat the process described above.

Getting related repository name

First of all, using GitHub API integrated application, using GitHub API, all repository name list in GitHub will be taken, if we want all repository in GitHub written in a particular programming language, we can retrieve using GitHub API by equate language parameter as particular language. For example, if we want all java repository name list in GitHub, language: java. So we can have all java project name list in this way. Indeed, if you take a project, it will contains many java classes. When we retrieve repository name list in GitHub, it will return in JSON format. As well as it will give repository name and author's name. After download all repository name list, we can have all java file name list (it is finished with .java extension) again using GitHub API. Now we have two repository such that, java repository name and .java file name. First we will compare the identifier with .java file name list, then we will get particular repository name from repository name list, and then the project will be downloaded.

GitHub API integrated application

GitHub provide an Application program Interface to researcher and developer for develop a system with GitHub API integrated application. When they do integration of GitHub API, the developers and researchers will face lot of technical problems. For these issues, GitHub maintain a support teams, they are servicing 24 hours, if you request solution to any problem faced in using GitHub API, within two hours they will give solution. Using GitHub API, we can do following things, All GIT activities within our application such that, Create, update, local changes, commit, branches and tag, etc. As well as we can have all repository details such as, repository name by language and by author, and class names. So we used GitHub API for last purpose.

Implementation

Developed tool, includes four main implementation. XML extractor, GitHub API integration, n-gram application, WordNet application. We will discuss each one by one.
3.1. XML Extractor.

Purpose of this class is extract information from XML file. For this purpose we used DOM (Document Object Model) parser, which is included in javax.xml package. It has two main classes such as, DocumentBuilder: Defines the API to obtain DOM Document instances from an XML document, DocumwntBuilderFactory: Define a factory API that enables applications to obtain a parser that produces DOM object trees from XML documents. As well as we used Document interface, this represents the entire HTML or XML document. Conceptually, it is the root of the document tree, and provides the primary access to the document's data. So by using those API, we developed XML extractor to extract information from XML file.

GitHub API integration

GitHub API was used for retrieve all java repository names list and all .java file names list. Normally, GitHub API provide those details in JSON objects. Using a JSON parser we can access those JSON string. When we use GitHub API, we will face lot of limitation and access level. I broke those limitations by using some neat techniques. The following limitation available in GitHub API such as, Even though one page contains more than million repository names, by default only 30 object will be displayed, as well as we can access only 30 object. But we can extend to 100 JSON object according to figure 2. After doing this this extend also we cannot access all JSON object. Then we can use particular date. We can access the projects which are created on particular date. Even though the number of project created on particular date were more than 5000. Then we used particular

time, so using one loop we can access projects which are created every hour in every day. For this we used calendar API in java.

URL using GitHub API	Details
https://api.github.com/search/repositories?q=language:java	It will return only 30 JSON objects in a page and number of projects in GitHub as well as details of the projects (author, created date, etc.)
https://api.github.com/search/repositories?q=language:java&per_page=100	It will return 100 JSON objects in a page and some details about the projects. But we access only 100 JSON objects
https://api.github.com/search/repositories?q=language:java+created:>2016-01-16&per_page=100	It will return all projects, which are created before 2016-01-16, now also we cannot access all projects.
https://api.github.com/search/repositories?q=language:java+created:2016-01-16T00:00:00Z..2016-01-16T01:00:00Z&per_page=100	It will return all projects, which are created on 2016-10-16 at 01:00:00.
https://api.github.com/search/repositories?q=language:java+created:2015-09-19T00:00:00Z..2015-09-19T00:00:01Z&per_page=100	It will return all projects, which are created on 2016-10-16 at 00:00:01.
https://api.github.com/search/code?q=repo:elastic/elasticsearch%20extension:java&start=0	It will return all .java file names within elasticsearch projects in GitHub, and on first page.

Figure2: URL to access all java repository names and all .java file names in GitHub and its details

Even we used each hour in a day, the number of project was more than 100, and then we split the query into seconds. The number of projects created in a seconds was less than 100. So we can access projects which are created in each seconds in each day.Another very big limitation is, we can make only 10 query within one minutes. Then we used a java threat to wait one minutes after make 10 query. GitHub API provides so many facilities, in which we used only repository search for all java repository name in GitHub, and code search for all .java file name list in a particular project.

Search API and repository search.

The Search API is optimized to help you find the specific item you're looking for (e.g., a specific user, a specific file in a repository, etc.). Think of it the way you think of performing a search on Google. It's designed to help you find the one result you're looking for (or maybe the few results you're looking for). Just like searching on Google, you sometimes want to see a few pages of search results so that you can find the item that best meets your needs.

Name	Type	Description
`q`	`string`	The search keywords, as well as any qualifiers.
`sort`	`string`	The sort field. One of `stars`, `forks`, or `updated`. Default: results are sorted by best match.
`order`	`string`	The sort order if `sort` parameter is provided. One of `asc` or `desc`. Default: `desc`

Figure3: The parameters used in Query

To satisfy that need, the GitHub Search API provides up to 1,000 results for each search**.**Unless another sort option is provided as a query parameter, results are sorted by

best match, as indicated by the*Score* field for each item returned. This is a computed value representing the relevance of an item relative to the other items in the result set. Multiple factors are combined to boost the most relevant item to the top of the result list.Find repositories via various criteria, this method returns up to 100 results per page. Figure 3 and figure 4 describe parameter need to use to get all java repositories in GitHub.

The q search term can also contain any combination of the supported repository search qualifiers.

Search qualifiers	Description
In	Qualifies which fields are searched. With this qualifier you can restrict the search to just the repository name, description, readme, or any combination of these.
size	Finds repositories that match a certain size (in kilobytes).
forks	Filters repositories based on the number of forks.
created or pushed	Filters repositories based on date of creation, or when they were last updated.
user or repo	Limits searches to a specific user or repository.
language	Searches repositories based on the language they're written in.
stars	Searches repositories based on the number of stars.

Figure 4: The search qualifiers used in query

Suppose you want to search for popular Tetris repositories written in Assembly. Your query might look like this.
https://api.github.com/search/repositories?q=tetris+language:assembly&sort=stars&order=desc. In above request, we're searching for repositories with the word “tetris” in the name, the description, or the README.

We're limiting the results to only find repositories where the primary language is Assembly. We're sorting by stars in descending order, so that the most popular repositories appear first in the search results. In this way we can break the limitations which we discussed in section3.2. But using these parameters and qualifiers we cannot break the number of request in a minutes.

N-Gram Application

N-Grams is a word prediction algorithm using probabilistic methods to predict next word after observing N-1 words. Therefore, computing the probability of the next word is closely related to computing the probability of a sequence of words. As we discussed in chapter 3.2 earlier, Using GitHub API, we make two repository. One includes all java repository name in GitHub, and all .java file name in GitHub. Now we have some identifiers form Class Diagram, as well as all class name from second repository (.java file names) from GitHub. But we need to do a comparing and matching process between two identifiers.

We use N-Gram technique to split a word to some grams. For an example, a word is “file”, if we consider two-gram then the grams are, “fi”, “il”, “le”. Using n-gram technique, we split an identifier into their grams, through this process, we make a

dictionary, which contains grams for all class name from our .java file names repository. When we take an identifier from class diagram, the n-gram technique will split the identifier to some grams, then do the matching process between the grams. If it matches, we retrieve the related repository name, and then we download the project. Otherwise we will use WordNet to retrieve some synonyms of an identifier.

WordNet Application
WordNet is a lexical database of English word and sense relations. A *sense* is a particular meaning of a word. For each sense of a particular word, WordNet provides the *synset*, a list of synonyms for that sense, provides short definitions and usage examples, and records a number of relations among these synonym sets or their members. WordNet can thus be seen as a combination of dictionary andthesaurus.

First we do a matching with two identifiers using n-gram technique, if it matches no problem, if it does not match, we need to get identifier's synonyms for further comparison, splitting those synonyms one by one, and do the matching process. To get synonyms of a word, we implemented the WordNet application in our project. For that, we use Java API for WordNet SearchingJAWS API. When we download this API, a dictionary will be included in the package.

Limitation and future work
This research is in progress, The GitHub API use is complex and crucial; it will give most relevant result than sourceforge.net and googlecode.com. That is why we publish this work. In this work we included only class diagram and we have done search only for java language. We can get most relevant project using GitHub API, but we cannot say, those project are accurately relevant, for more relevant we are going to use some more NLP technique such as Part of Speech tagging, and n-gram technique for result.

Incorporating the technique Dynamic Time Warping is planned to explore in future. In this technique, the keywords which is extracted from Software architecture and extracted from Source code will be put on an x-axis on a graph as vector and some related word (synonyms) from a lexical dictionary (WordNet) will be put on y-axis. Then it will calculate a DTW distance for all words, smaller distance words will be selected. In this way we can split the word which is cannot identify by our approach discussed in this paper.

We will do some comparison between those identifier and most relevant one will be selected. In this way we will suggest some related sample code or related project to programmer. As well as we planned to automatically suggest some Design pattern for those project. For suggesting Design pattern, we planned to use Case Based reasoning, machine learning technique to suggest suitable design pattern for the project along with suitable sample code.

Conclusion

Code reuse is complex and crucial especially for small company without an expert in a particular area. We have developed a tool integrated with GitHub API, an approach for addressing problems faced by programmers in reusing existing frameworks or libraries or sample project in GitHub. Our approach will start from a Software architecture, by importing a Software Architecture (class diagram) as XML format file, and then extract some information from the XML file. Depend on the information extracted from XNL

file (Identifiers), it will compare the information with the .java file names lists from our premade repository using GitHub API using n-gram technique and WordNet.

If an identifier matches with any file name (class names of java files), related project name will be retrieved from the premade java repository names list database from the GitHub. Then the particular project's master repository will be downloaded. These project will be most relevant projects to the Software Architecture. This is most relevant but not actual related project, for actual related project, in further development, we will use speech recognition technique such that DTW.

References

[1]. Lars Heinemann, Prof. Dr. Dr. h.c. Manfred Broy Prof. Martin Robillard, McGill University, Montréal, Kanada, "Effective and Efficient Reuse with Software Libraries"presented at the20th ACM SIGSOFT International Symposium on Foundations of Software EngineeringJully 2012

[2]. Madhuri R. Marri, Suresh Thummalapenta, Tao Xie,Department of Computer Science North Carolina State University, "Improving Software Quality via Code Searching and Mining," presented at the. ICSE Workshop and conference, Vancouver, British, Columbia, May 2009.

[3]. Sushil K Bajracharya, Joel Ossher, and Cristina V Lopes,"Leveraging Usage Similarity for Effective Retrieval of Examples in Code Repositories". Presented at ACM SIGSOFT international symposium on Foundations of software engineering, University of California, Irvine, CA, USA November 2010.

[4]. NiooshaMadani, LatifaGuerrouj,; "Recognizing Words from Source Code Identifiers using Speech Recognition Techniques", presented at the CSMR 2010 14th European Conference at Madrid, Spain, March 2010.

[5]. MiltiadisAllamanis, Christian Bird,"Learning Natural Coding Conventions", presented at 22nd ACM SIGSOFT International Symposium on Foundations of Software Engineering Microsoft Research, Microsoft, Redmond, WA, USA, November 2014.

[6]. ApostolosKritikos, George Kakarontzas and IoannisStamelos. "Asemi-automated process for open source code reuse", Thessaloniki, Thessaloniki, Greece, presented at International Conference on Evaluation of Novel Approaches to Software Engineering 2010.

[7]. Matthew J. Howard, Samir Gupta, Lori Pollock, and K. Vijay-Shanker, "Automatically Mining Software-Based, Semantically-Similar Words from Comment-Code Mappings". Presented at 10th Working Conference on Mining Software Repositories, May 2013.

[8]. GiriprasadSridhara, Emily Hill, Lori Pollock and K. Vijay-Shanker, "Identifying Word Relations in Software:A Comparative Study of Semantic Similarity Tools," presented at the 16th IEEE International Conference, Amsterdam, Netherlands, June 2008.

[9]. Samir Gupta, Sana Malik, K. Vijay-Shanker, "Part-of-Speech Tagging of Program Identifiers for Improved Text-based Software Engineering Tools", presented at Program Comprehension (ICPC), 2013 IEEE 21st International Conference May 2013.

[10]. Tao Xie, Suresh Thummalapenta,"PARSEWeb: A Programmer Assistant for Reusing Open Source Code on the Web", presented at twenty-second IEEE/ACM international conference November 2007.

[11]. Steven P. Reiss, Department of Computer Science, "Specifying What to Search For", Presented at IEEE conference at Brown University, May 2009.

[12]. Adrian Kuhn, Florian S. Gysin, "A Trustability Metric for Code Search based on Developer Karma" presented at ICSE Workshop on Search-driven Development: Users, Infrastructure, Tools and Evaluation, May 2010.

[13]. Otávio Augusto LazzariniLemos, SushilBajracharya, Joel Ossher, Paulo Cesar Masiero and Cristina Lopes, "Applying Test-driven Code Search to the Reuse of Auxiliary Functionality". Presented at 2009 ACM symposium on Applied Computing, March 2009.

[14]. Emily Hill, Lori Pollock and K. Vijay-Shanker, "Automatically Capturing Source Code Context of NL-Queries for Software Maintenance and Reuse", presented at ICSE '09 Proceedings of the 31st International Conference on Software Engineering, May 2009.

[15]. Steven P. Reiss Department of Computer Science, Brown University, "Semantics-Based Code Search", presented at ICSE '09 Proceedings of the 31st International Conference on Software Engineering, may 2009.