iJRASET
International Journal For Research in Applied Science and Engineering Technology



INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 5 **Issue:** II **Month of publication:** February 2017

DOI: http://doi.org/10.22214/ijraset.2017.2021

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)

Extracting Development Tasks to Navigate Software System

Mrs. Yele Kiran S. [1], Mrs.Karnwar Pravin B. [2], Mrs. Patil Ulash C. [3], Vedpathak Ajit D. [4], Mrs. Thorat Sanjay D. [5]
Department of Information Technology, SVPM's College of Engineering, Malegaon (Bk), Tal-Baramati, Dist-Pune, Savitribai Phule Pune University.

Abstract: ***Many software development organizations and open-source projects attempt to address this challenge by creating web pages that collect the most important information.. Whenever developer build a software, he struggles to find the right information in the right form at the right time so to help developers navigate documentation. In this paper we survey various techniques for extracting structured and unstructured data and automatically extracting tasks from software documentation by conceptualizing tasks as specific programming actions that have been described in the documentation.This essential information need to be extracted and annotated automatically which is challenge in software engineering.In this paper we develop Task Navigator a user interface search for queries that suggests tasks extracted with our technique in an auto complete list along with concepts, code elements and section headers.***

Keywords: Software Documentation, Development Tasks, Data Extraction, Retrieved Task, bookmark, Navigation, Auto-Complete, Natural Language Processing.

I. INTRODUCTION

A software developer or other any stakeholder who joins an existing software development team must come up to speed on a large for varied amount of information before becoming productive. Today's all types of documentation suffers from a number of potential problems such as documentation written by people who can't write.They are unavailable, developer can't find it when they need it. Today's search engines are not sufficient for enabling effective navigation of software documentation because they require stakeholder to use search terms that match the vocabulary used by the documentation writers and documentation may be in the form of soft copy of project or hardcopy.There is still a gap between the information needs for software developers and the structure of today's documentation. Today's structure comes with sections and subsection it can only be enabled effective navigation if the section headers are adequate for the information needs of developers. We divided our main project with different module such as Pre-processing, Task Extraction and Task Navigation techniques used in the software development industry. In short software,Extracting Development Tasks To Navigate Software Documentation documentation enlists enough and necessary requirement that are required for the project development, but there is still a gap between the information needs of software developers and the structure of this documentation. Any kind of structure with sections and subsections can only be enable effective navigation if the section headers are adequate clues for the information needs of developers.

A. Problem statement

To provide the necessary and appropriate information to a user as a baseline In many software development company their are certain problem to finding exact data or information at right time data related to that developer working on that particular Domain software task or module. While much of the important technical knowledge can be captured in documentation, there often exists a gap between the information needs of software developers and the documentation structure. we plan to over Task Navigator to open source projects, and we aim to improve the precision of the task extraction.

B. Objectives

We evaluated the accuracy of the preprocessing steps and the task extraction algorithm using a benchmark of sentences and their corresponding task. We compared the relevance of the task-based auto-complete suggestions to the relevance of auto-complete.
To evaluate whether the extracted tasks are meaningful to developers, we conducted an evaluation of the tasks extracted from the documentation of two projects with 10 professional software developers.

II. PROPOSED SYSTEM

A. System Description

©IJRASET: All Rights are Reserved

Document extraction: In Document extraction is the client server application where client interact with server through the web browser.
Web Server: In web server At web server will receive client request for document search.
User Interface: In user interface To send request and get response back we design user interface with HTML, JSP, and SERVLET.
Preprocessing: In this preprocessing We divide our application with different phases such as preprocessing, Task Extraction and Task Navigation.
User: In user phase While user ask query for document extraction server check keyword and phrases by extracting concepts from search query.
Indexing: In Indexing Indexer will look for reference file from page history in document repository whether reference is exist in database. If exist return page to recommendation for client.
Reverse Engineering: In reverse engineering phase server will extract document if index exist and if not exist preprocess the document.
Task Navigation: In Task Navigation phase execute algorithm for search document with index number. Navigate through documentation for expected page.
Extraction: In Extraction phase will extract document from repository. It returns response with index of that page to server.
Recommendation:
In Recommendation phase fetch document according to index and show intelligence to navigate through document.
Search Interface: In this Search Interface shows auto complete search for document in User Interface.

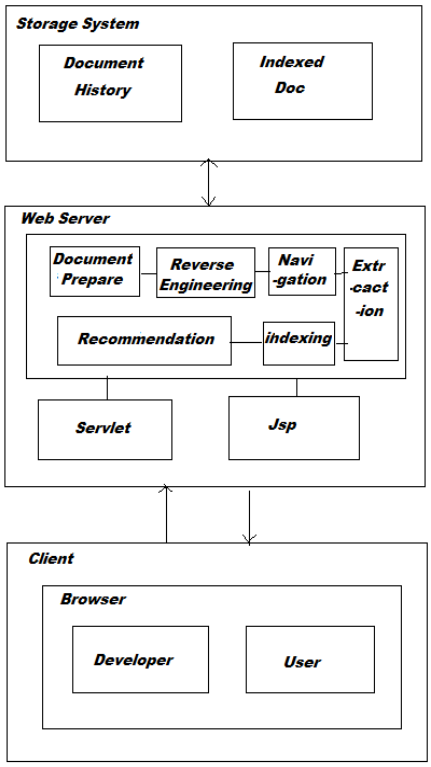
B. Mathematical Model

S = {s, e, X, Y, F,NDD,Ø}
Where,
s = Start of the document extraction system/child Tracking System
Connect with Application.
Deploy the Application to Server.
End of the System.
suggestion from document repository.
X = Input of the program.
Keywords. Phrases.
Index value for search.
Output of the program.
Discovering meaningful new document, provide Recommendation of package to
developer or user.
X, Y "U
Let, U be the Set of System.
U = {M, extractor, indexer}
M=Documents repository.
Extractor= Document Extractor.
Indexer=Show navigation index towards document.
F = {F1; F2; F3; F4} Where,
F1= Document preparation.
Extracting Development Tasks To Navigate Software Documentation
F2= Assigning index.
F3=Extracting Task from document.
F4= Navigation on Document by index hierarchy.
DD= since F gives expected result, therefore problem is deterministic.
Ø= Failures and Success conditions

C. System Architecture

©IJRASET: All Rights are Reserved

Volume 5 Issue II, February 2017
ISSN: 2321-9653
International Journal for Research in Applied Science & Engineering Technology (IJRASET)



D. Advantages

Automatically extracting tasks from software documentation.
To evaluate whether the extracted tasks are Meaningful and Accurate.
Efficient when used on large systems, Tester can be non-technical.
Easy to handle User friendly system.
Easy to Data Retrival, User can Get approximately and needed data.

III. MODULES

A. Extracting Development Task

1) *Preprocessing:* In this preprocessing to enable the extraction of development tasks and the documentation corpus of a project is preprocessed by transforming HTML files into text files. In this phase most of the HTML mark-up is removed while keeping the linebreak information and then redundant information that is repeated on each page of the documentation.
2) *Task Extraction:* In this phase, here we define a task in software documentation as a specific programming action to describe in the documentation.
3) *Concepts:* In this phase as a baseline for assessing the usefulness of development tasks for navigating software documentation.
4) *Code Elements:* In this phase by using preprocessing step the extraction of code elements is straightforward. In addition to concepts, we extract code elements from documentation.

B. Search Interface

Index Entries In this phase each index entry is an instance of a documentation Element. Eg. a task, a concept, a code element. tasknavigator uses as input a set of index entries.

C. Accuracy of the Algorithm

1) *Accuracy of the Task Extraction:* To evaluate the accuracy of the task extraction algorithm.
2) *Evaluation:* To evaluate whether the extracted tasks are meaningful to software developers.

©IJRASET: All Rights are Reserved

IV. CONCLUSION

We also evaluated task-based navigation with a field study in a corporate environment,in which six professional software developers used the tool for two weeks as part of their on going work. We found search results identified through development tasks to be more helpful to developers than those found through concepts, code elements, and section titles. These results indicate that development tasks can be extracted from software documentation automatically, and that they can help bridge the gap between software documentation and the information needs of software developers.

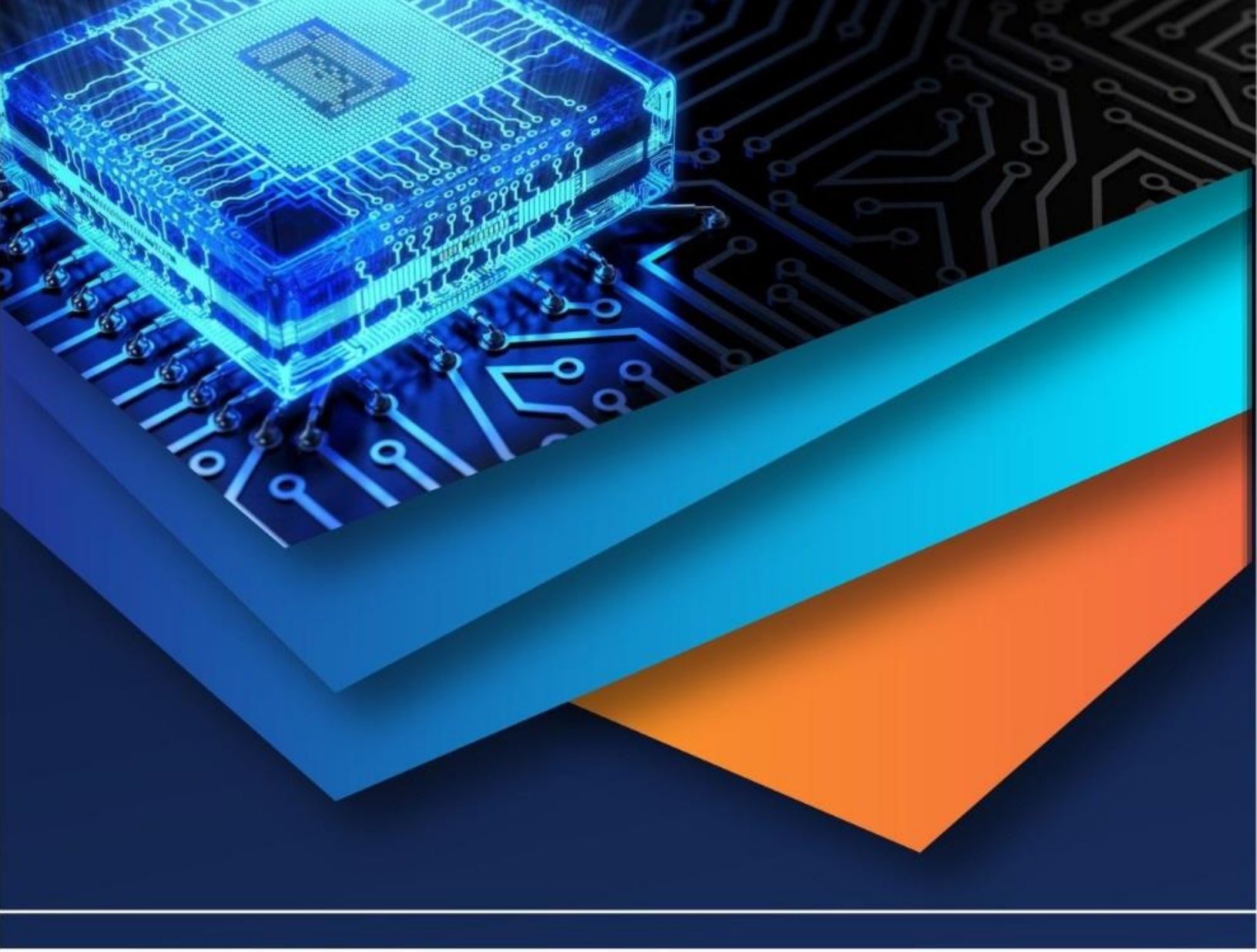
V. ACKNOWLEDGEMENT

A project of this magnitude has been a journey with various ups and downs. It was the support from Guide, Colleagues and family, which has helped me in the successful accomplishment of this project.

REFERENCES

[1] S. L. Abebe and P. Tonella. Natural language parsing of program element names for concept extraction. In Proceedings of the 18th IEEE International Conference on Program Comprehension, pages 156–159, 2010.
[2] G. Antoniol and Y.-G. Gu´eh´eneuc. Feature identification: A novel approach and a case study. In Proceedings of the 21st IEEE International Conference on Software Maintenance, pages 357–366,2005
[3] A. Bacchelli, A. Cleve, M. Lanza, and A. Mocci. Extracting structured data from natural language documents with island parsing. In Proceedings of the 26th International Conference on Automated Software Engineering, pages 476–479, 2011.
[4] Colin J. Neill and Phillip A. Laplante, “Requirements Engineering :The State of the Practice,” IEEE SOFTWARE Published by the IEEE Computer Society, pp. 40–45, Dec 1955.
[5] Vivek D. Mohod and Mrs. J. V. Megha, "A Survey on Data Extraction of Web Pages Using Tag Tree Structure,"IJCSIT International Journal of Computer Science and Information Technologies, pp.4361-4363, Jan 2005.
[6] Hubert F. Hofmann and Franz Lehner, "Requirements Engineering as a Success Factor in Software Projects,"IEEE SOFTWARE Published by the IEEE Computer Society, pp. 58–66, July 2001.
[7] Girish K. Palshikar and Rajiv Srivastava, "Information Extraction for Effective Knowledge Management"TCS White paper, pp. 1-14.
[8] Christoph Treude and Mathieu Sicard, "TaskNav: Task-based Navigation of Software Documentation",IEEE/ACM 37th IEEE International Conference on Software Engineering, pp. 649-652, June 2015.

©IJRASET: All Rights are Reserved



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Call : 08813907089 (24*7 Support on Whatsapp)