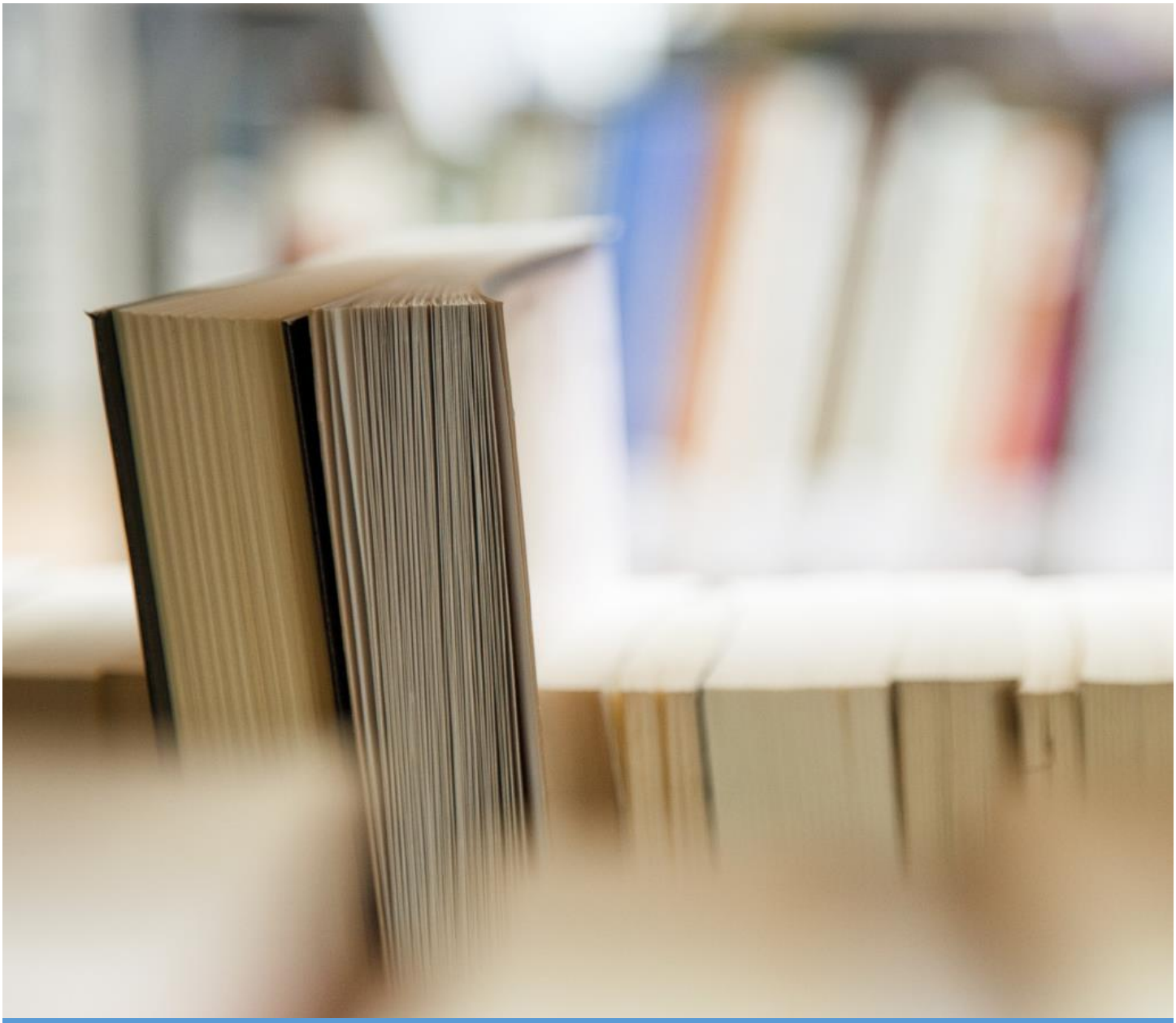


Project Proposal

M. A. Nur Quraishi

8/15/17

SE-801

Project Name:

"Recovering Documentation to Source Code Traceability Links using Latent Semantic Indexing"

Supervised by-

Rayhanur Rahman
Lecturer at
IIT, DU

Submitted by-

M. A. Nur Quraishi
Roll: BSSE-0615
BSSE 6th Batch
IIT, DU

Table of Contents

Table of Figures

1 Introduction

First of all, I want to clarify that I am going to implement an existing work or method here. It is analyzed by Andrian Marcus and Jonathan I. Maletic from Department of Computer Science, Kent State University, Kent Ohio.

The whole software engineering community (both research and industrial) in this era step forward to improve the explicit connection between system documentation and application source code. Lots of integrated environment and CASE tools are developed for solving this issue. These tools and techniques have played a major role to identify the traceability links from documents to source code and develop a new software system. Regrettably, many of these methods are unable to apply on the existing or legacy system. Recovering Documentation to Source Code Traceability Links (RDSCTL) using Latent Semantic Indexing is one of these methods which is pretty effective, highly flexible and of low cost.

1.1 What is RDSCTL?

Recovering Documentation to Source Code Traceability Links (RDSCTL) [1] is an information retrieval mechanism that uses Latent Semantic Indexing (LSI) to automatically generate or identify the traceability links from system documentation to program source code.

1.2 What is LSI?

Latent Semantic Indexing (LSI) [2,3] is a Vector Space Model (VSM) based method for generating and representing aspects of the meanings of words and passages reflective in their usage. It is also used in natural language processing (NLP).

In NLP, LSI not only grabs important portions of the meaning of individual words but also of whole passages such as sentences, paragraphs, articles and short essays.

The main concept of LSI [1] is that the information about word contexts in which a particular word appears, or does not appear, provides a set of mutual constraints that determines the similarity of meaning of sets of words to each other.

2 Motivation for the Research

As I mentioned before the whole software engineering community working on it to find out the explicit connection between documentation and source code. Identifying the traceability link between these two terms or factors in a legacy system is particularly important for various software engineering tasks. These include [1]:

- general maintenance tasks
- impact analysis
- program comprehension
- more encompassing tasks such as reverse engineering for redevelopment and systematic reuse

Moreover, recovering documents to source code traceability link is very important for academic and industrial purpose too. Through these traceability links a client can measure his/her satisfaction rate by comparing his/her requirements and the product (source code) in hand. Similarly, it is also important for academic purpose as a teacher can measure or estimate the actual implementation rate that students stated or promised in their SRS. Thus, a teacher can identify which

feature is missing from the source code or application which is previously promised by a student.

There are lots of techniques and methods to find out the traceability links and retrieve information.

2.1 Why RDSCTL is better?

One of the distinct advantage of our method is that it does not rely on predefined vocabulary or grammar for the documentations or source code which allowed the method to be applied without large number of preprocessing or manipulation of the input that drastically reduces the cost and time of link recovery.

2.2 Why LSI is used in RDSCTL?

A common criticism of VSM is [1] that it does not consider the relations between terms. For example, having "automobile" in one document and "car" in another document does not contribute to the similarity measure between these two documents. VSM shows zero similarities between documents that share no terms.

LSI attempts to overcome this shortcoming by choosing linear combinations of terms as dimensions of the representation space.

3 Assumptions

The assumptions for RDSCTL are:

- the comments and identifiers are reasonably named
- developers use the same natural language (e.g., English, Romanian, etc.) in writing internal documentation (source code) and external documentation (SRS)

4 Problem Statement

The work presented in the paper that has been followed addresses two specific problems:

i. using Information Retrieval (IR) methods to support software engineering tasks
ii. recovering source code to documentation links.

5 Research Methodologies

Figure 5.1 depicts the major elements in the traceability recovery process. The entire process is designed in a pipeline architecture. The output of one phase becomes the input for the next phase. The user's involvement in the process occurs in the beginning for selecting the source code and documentation files. Then the user selects the dimensionality of the LSI subspace. After the LSI subspace is generated, the user determines what type of threshold will be used in determining the traceability links.

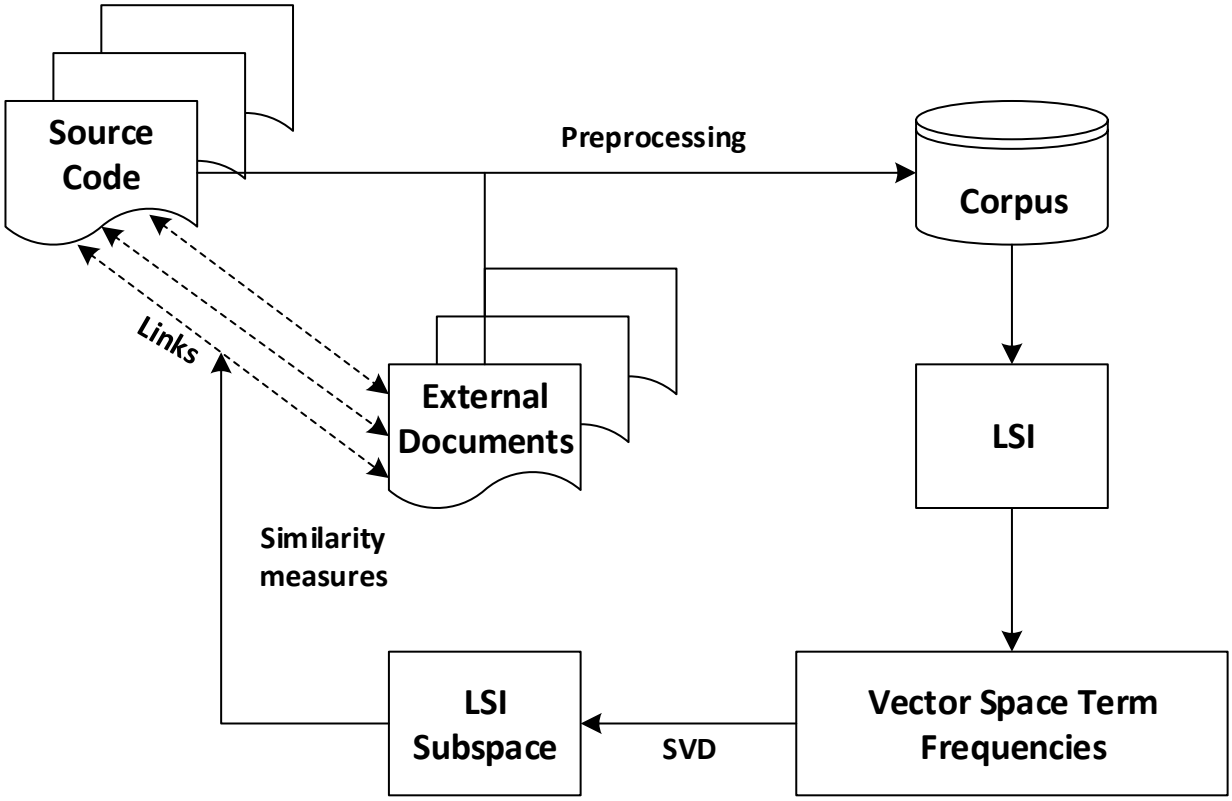


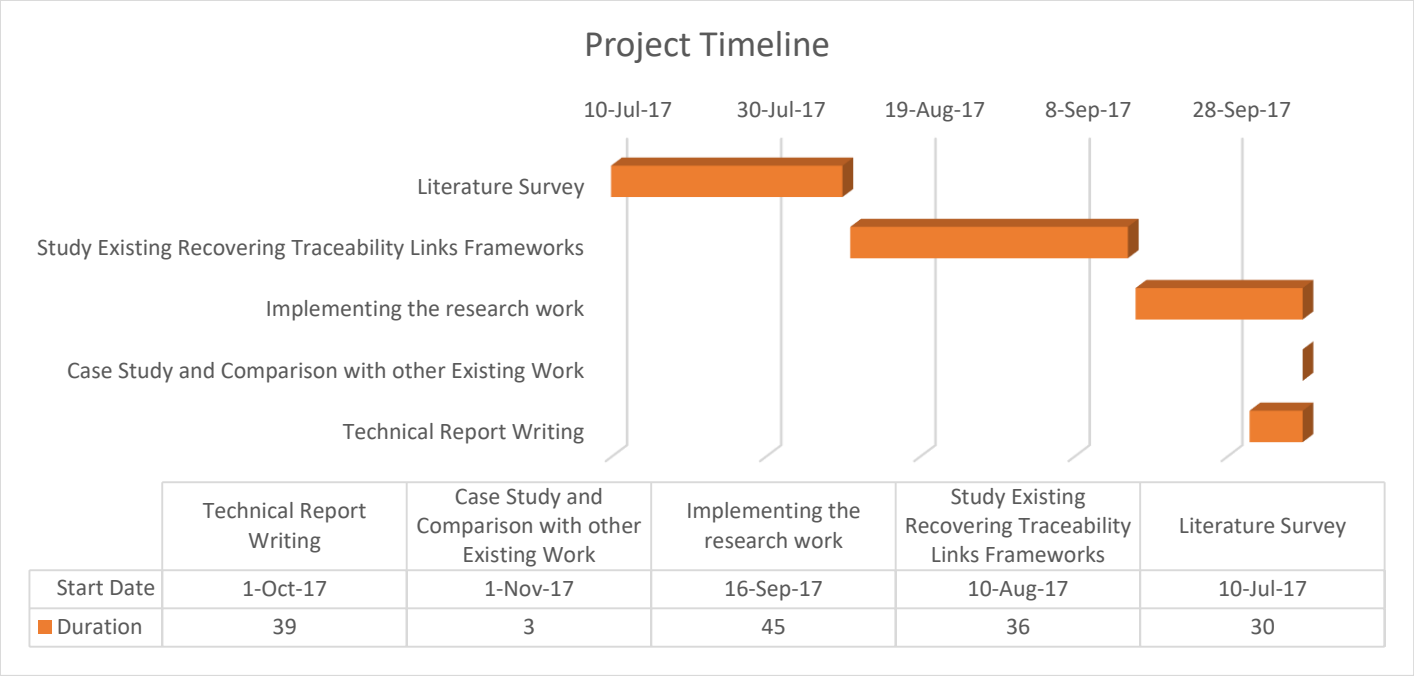
Figure 5.1 RDSCTL Process

Initially, I went through some literature study to identify the area of interest and the scope of contribution in those areas. After that, I Have selected the appropriate section where improvement can be done in future and set the goal of research. Then I have started background study related to this task which is a continuous process and will go through the whole semester. At the same time, I have prepared a research proposal and submit it.

Then I will study the existing framework. The next activity will be to convert the hypothetical idea into a framework. For that, at first the system documentation and source codes need to be provided as input. Then the documents need to be preprocessed for further work. Then the necessary information needs to be extracted and used for recovering the traceability links from documents to source code. After the

implementation of the framework, it needs to be evaluated and compared with other existing frameworks.

Step No.	Title of Activity	Activity description
1.	Literature Survey	• Building the background for the research. This will be a continuous task. • Writing research proposal.
2.	Study Existing Recovering Traceability Links Frameworks	• Intensifying the focus on more specific area. • This task will cover the similar or related works to our area of interest exhaustively. • Reporting on survey.
3.	Implementing the research work	• The framework will be generated here.
4.	Case Study and Comparison with other Existing Work	• Studying existing frameworks to identify the required structure for Recovering Traceability Links. • Evaluating the framework against other existing ones. • Producing a Technical report on the findings.
5.	Technical Report Writing	• At the end, one technical report will be published.



	Technical Report Writing	Case Study and Comparison with other Existing Work	Implementing the research work	Study Existing Recovering Traceability Links Frameworks	Literature Survey
Start Date	1-Oct-17	1-Nov-17	16-Sep-17	10-Aug-17	10-Jul-17
Duration	39	3	45	36	30

Figure 5.2 Project Timeline

6 References

[1] Andrian Marcus, Jonathan I. Maletic, “Recovering Documentation-to-Source-Code Traceability Links using Latent Semantic Indexing”

[2] Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., and Harshman, R., "Indexing by Latent Semantic Analysis", Journal of the American Society for Information Science, 41, 1990, pp. 391-407.

[3] Dumais, S. T., "Improving the retrieval of information from external sources", Behavior Research Methods, Instruments, and Computers, 23, 2, 1991, pp. 229 - 236.