Volume 4, Issue 2
April 2015

ITEE Journal
Information Technology & Electrical Engineering

ISSN: - 2306-708X

A Comparative Study on Traceability Approaches in Software development Life Cycle

Muhammad Hassnain
Abasyn University Islamabad Campus
Islamabad, Pakistan
Email: hassnain.pk77@gmail.com

ABSTRACT

Traceability supports the software developers to keep the track record of software changes during the software artifacts developments. Most recent research publications are included to compare the traceability approaches in terms of a defined set of seven parameters. Each of the research study has covered at least a set of pair of features that reveals their importance in software artifacts development. This comparative analysis work is mainly focused on case studies used in the research works. Anti-lock braking system and microwave oven product line are two important case studies, which involved the traceability applications in the system development. Traceability features range from coverage type to coverage-based reconstruction that distinguishes the research works from each other.

Keywords— Traceability, Traceability Matrixes, Comparative Analysis, Paths, Links.

1. INTRODUCTION

Traceability is the capability for describing and following the artifacts of a software life, and describes the links that connect the associated artifacts. Traceability assists the software engineers in the effective development and management of software systems. Inadequacy in traceability results into budget overruns and project failures and leads to a low software maintenance. Traceability supplements the testing and debugging phase and developers locate the errors in less time and rectify these errors more reliably. Recovery in traceability link has attracted the attention of a large number of researchers in production of large and complex software systems. In this regard, re-establish the traceability relation between varying artifacts has been remained the focus of researchers in developing the new techniques to retrieve the traceability links.

This paper has been organized as follows. Section II looks at various related techniques on traceability approaches in software development life cycle from previous researches. In the next section III comparative analysis of identified approaches in perspective of various parameters has been described. In discussion section IV the main points derived from the previous section on comparative analysis are further discussed to reach a point about most favorable traceability techniques.

2. LITERATURE SURVEY

In a study of Rempel et al., found that requirement traceability helped the developers in attaining the good quality products with a higher maturity level. Researchers called this requirement traceability an effective approach that used efforts and never occurred through a chance or ad-hoc. They also added that traceability was defined upfront as explicitly. Planned requirement traceability was more supportive in form of a strategy in order to support the developers in software life cycle. Practitioners from 17 companies presented their views on traceability paths, artifacts, traceability usage, traceability usage model, advantages of identified traceability paths, identification of and missing trace paths. After identification of traceability missing paths and goals a set of superfluous traceability goal was achieved [1]. Among these facts trace path, usage goals and their suitability met with issues of ensuring the long existence of traceability path and tasks not covering any goal respectively. Some of the ambiguous artifacts were also identified from cases used in their study. Upfront defined traceability as a strategy could be used for determining the complex tasks about traceability paths for a system. It was also found that a well traceability strategy defined must be clearly defined and assessed according to goal-driven principles.

Reliability level of software maintenance is determined through the key activity of testing. Modeling between testing and debugging is most effectively facilitated by using the traceability information. Detailed literature review was conducted to distinguish the traceability requirements and traceability. Functional and non-functional traces were significantly revealed. Functional traces were transformed from one artifact to another artifact by using a set of defined rules. These functional traces are explicit traces, which are either created as a by-product or reconstructed unambiguously. On the other hands, non-functional traces are

ITEE Journal
Information Technology & Electrical Engineering

ISSN: - 2306-708X

described in the context of reasons, decision, context and technical. Creative process results into non-functional traces, which are analyzed semantically and extracted from meetings with the customers [2].

Zhou in a study [3] introduced the IR-based traceability recovery for requirement management. Requirement traceability has been called as heart of requirement management. Under this approach, information retrieval process compares the source artifacts (requirements) and target artifacts (source code files). Similar texts from pair of artifacts as in form of terms are contained in the repository. Paired artifacts with similar texts are filtered out and form the candidature of traces of link list. The IR-based traceability approach shows that artifacts are preprocessed before they are computed for similarities. Furthermore, operators, stop words, keywords, articles and punctuations in the text are discarded. Extracted terms, after preprocessing are stored as m x N in a matrix where m represents the number of terms in a document and N is the number of documents.

In another approach in [4] Marques et al., integrated the requirement management and Model driven engineering into engineering process of embedded systems. A case study based on the control system was embedded into the anti-lock braking software along with its requirements management. Integrated model was applied to generate the traceability matrix. Required system was a break system, which prevented the wheel lock-up that avoided the traction loss and control needed during the emergency. This system was based on the requirement phases including the elicitation, analysis and specification, validation and requirement management. Design traceability was supported by the information retrieved from use case "check speed" that associated to requirement with the same relationship. Traceability matrixes were too involved in management activities. Matrixes were created automatically from requirement diagram. Dependencies between requirement to design artifacts or test cases were captured from traceability matrixes. Following this traceability approach, it produced real embedded systems as well as the management of system requirements.

Kang et al., presented the traceability in a way of formal representation of requirements. Traceability was called as an important enabler of developers to keep the changes' track for system development artifacts. Although, traceability management has been more complicated process as compared to single product development but deals with the traceability of a single product as well as variability and commonality found among the product families. Existing traceability approached worked as an ad-hoc for software product families and did not address the link between domain artifacts and application artifacts in the relative domain. In new approach [5] constraints for domain engineering and product are satisfied. Domain engineering process constructs the DE artifacts. Approach provided more effective management of changes during the software development lifecycle.

3. COMPARATIVE ANALYSIS

This section gives the comparative analysis of identified traceability approaches based on some of important criteria such as coverage types, Software traceability support, traceability matrixes, IR-based, goal centric, rule based and coverage-based reconstruction traceability approach.

- **Coverage Types (CT):** Effective coverage determination and analysis tells that which type of coverage type is used for the relative traceability approach. There are number of coverage types that researchers use including the requirement, path, statement, methods, design and functions.

- **Software Traceability Support (STS):** Software traceability determines the links between software artifacts (requirement, code and test cases) in either way of requirement traceability and code traceability. In many approaches the trend of code-to-test and then measure test coverage are widely used.

- **Traceability Matrixes (TM):** Traceability matrix shows the correlation between any two or more than two documents. Traceability matrix is widely applied for high-level requirements and detailed requirements for a system that matches to a high-level design parts, test plan, design and test cases.

- **IR-based (IrB):** An IR-based approach traceability approach is focused on the creation of automatic traceability links from comparison of two different types of artifacts.

- **Goal Centric (GC):** In fact, goal centric traceability approach has been developed for maintaining the non-functional requirements. Because, NFRs' handling is more complex as compared to functional requirements because NFRs are based on a specific criteria that estimates a system's operations and specific behavior. A goal centric approach supports the stakeholders for impact evaluation caused by the changes in functional requirements to non-functional requirements.

- **Rule-based (Rb):** Rule-based traceability approach encompasses the requirement statement and use cases documents. It also covers the object models for analysis as an object of traceability. By using rules, certain types of traceability links are automatically

ITEE Journal
Information Technology & Electrical Engineering

ISSN: - 2306-708X

created. These traceability rules are deployed on the specific types of documents including the use case documents, requirements statement documents and object model of analysis document.

- **Coverage-based Reconstruction (CbR):** Change in a software like method, class, test case, requirements and design changes lead towards the changes in codes. Therefore, coverage reconstruction has a significant role in the coverage analysis.

Using the above parameters, traceability approaches have been evaluated. Rational behind choosing such criterion is whether identified approaches are compatible and capable in mapping among the artifacts during the SDLC. Artifacts include the documents of requirement specification, system design, code and test suits. Comparison among all approaches shows that a link between requirements artifacts to source code to a maintenance level exists.

This is a general criterion that is mostly used for evaluation of traceability approaches in the software development life cycle. Main aim is to identify the concerns and advantages of traceability approaches developed in recent few years.

Score of each research publication against given seven parameters has been shown in table 1. A traceability approach that addresses the types of coverage, and traceability support for a system development uses traceability matrixes. These approaches also show the relationship between artifacts of software development. Score for all approaches in terms of software traceability support is same that shows all approaches are centered to support the system development.

Research	CT	STS	TM	IrB	GC	Rb	CbR
P. Remp et al., [1]	Yes	Yes	No	No	Yes	No	No
Reza Meimandi Parizi, *et al.*, [2]	No	Yes	No	Yes	Yes	No	Yes
J. Zhou [3]	Yes	Yes	Yes	Yes	No	No	No
M.R.S Marques et al., [4]	No	Yes	Yes	No	Yes	No	No
S. Kang et al., [5]	No	Yes	No	No	No	Yes	No

Table 1: Comparative Analysis of Traceability Approaches

4. DISCUSSION AND ANALYSIS

Even though, this paper is focusing upon the traceability approaches during software development life cycle, but most of the identified traceability approaches are applied in different areas of software development. These areas range from requirement to coding and then implementation. This work brought insight that how traceability approaches were used in phases of software development life cycle.

During this work, some advantages and weaknesses of all traceability approaches are mentioned. Comparative study of traceability approaches has been worked on a medium level. It is a good beginning for enhancing my interest to resolve the specific issue about applications of traceability model. It is clear that all traceability approaches [1], [2], [3], [4], [5] are mainly focused on software traceability support. All of the traceability approaches are aimed to support the stakeholders involved in the software development. Studies [1], [3] showed the coverage types usage for relative traceability approaches.

In [1] trace path as a coverage type was considered and found that if path traces were not stored with determination then trace usage and long term existence of traces cannot be ensured. Some of the traces paths have same semantics and presented the redundancy problem. Most of studies used the traditional methods for requirement validation for functional as well as non-functional requirements. Studies [3], [4] showed usage of matrixes from the extracted terms for documents and artifacts. Matrix is also used for identification of a modified requirement that links to an artifact. Among traceability approaches, informational retrieval methods are important for software development as well as maintenance. This information retrieval as used in [2], [3] that has proven an advantageous to establish the traceability links in a short duration. Goal centric traceability approaches are mainly focused on management of non-functional as well as functional requirements.

Except research work [5] all other research studies have not involved the rule based traceability approaches. Same research work is advantageous over others as a platform feature to requirement traceability has been constructed. A case study is conducted on microwave oven product line. Product features as inputs, requirements, planned traceability and finally output are performed. Planned set of products along with a set of rules are probably used for development of a new product. This study is limited as it focuses upon the traceability from feature to requirements. Coverage base

construction is the fundamental feature in [2] that shows change in software code is directly linked with changes in requirements, methods, test cases and design.

5. CONCLUSION AND FUTURE WORK

In this paper, comparative analysis of traceability approaches has been conducted depending upon seven important features. It is seen that not a single study covers all areas taken in this comparative research paper. Efforts are made to cover importance of traceability approaches ranging from requirement phase to implementation phase. In selected research papers, it is observed that how traceability enables the links between software artifacts.

In the future study, more specifically forward and backward tracing between requirements and design phases will be focused. Therefore, this research study is preliminary start for a comprehensive research in the future.

REFERENCES

[1] P. Rempel, *et al.*, "An empirical study on project-specific traceability strategies " in *Requirements Engineering Conference (RE), 2013 21st IEEE International* Rio de Janeiro 2013, pp. 195-204.

[2] Reza Meimandi Parizi, *et al.*, "Achievements and Challenges in State-of-the-Art Software Traceability Between Test and Code Artifacts," *IEEE Transactions on Reliability*, vol. 63, pp. 913-926, 2014.

[3] J. Zhou, "Requirements Development and Management of Embedded Real-Time Systems," in *Requirements Engineering Conference (RE), 2014 IEEE 22nd International* Karlskrona, 2014, pp. 479-484.

[4] M. R. S. Marques, *et al.*, "Integrating UML, MARTE and sysml to improve requirements specification and traceability in the embedded domain " in *Industrial Informatics (INDIN), 2014 12th IEEE International Conference* Porto Alegre 2014, pp. 176-181.

[5] S. Kang, *et al.*, "A Formal Representation of Platform Feature-to-Requirement Traceability for Software Product Line Development " in *Computer Software and Applications Conference (COMPSAC), 2014 IEEE 38th Annual* Vasteras 2014, pp. 211-218.

AUTHORS' PROFILES

Muhammad Hassnain is currently doing his MS(CS) from Abasyn University Islamabad Campus Pakistan.