Tool Support for Traceability of Software Artefacts

K. Kamalabalan, T. Uruththirakodeeswaran, G. Thiyagalingam, D. B. Wijesinghe, I. Perera, D. Meedeniya
Department of Computer Science and Engineering, University of Moratuwa, Sri Lanka

D. Balasubramaniam
School of Computer Science, University of St Andrews, UK

***Abstract*—Artefact management in a software development process is a challenging problem. Often there is a wide variety of artefacts, which are maintained separately within a software development process, such as requirement specifications, architectural concerns, design specifications, source codes and test cases, which are essential to software engineering. Artefact inconsistency is a major problem since these artefacts evolve at different rates. Maintaining traceability links among these artefacts and updating those artefacts accordingly can be a solution to address artefact inconsistency. There is a need for establishing these artefact traceability links in semi-automatic way. Proper management and visualization tool is required for effective software artefact management in an incremental software development. We present a prototype tool to establish artefact traceability links and visualization of those. This paper describes the research methodology and relevant research carried out for semi-automatic traceability link establishment and visualization of software artefacts.**

Keywords—Artefacts; Traceability Links; Artefact Traceability Visualization; Artefacts Consistency Management

I. INTRODUCTION

Software development lifecycle consists of different process that result in different artefacts such as requirements specification, software architecture, design specification, source code, test cases for verification and validation of the system, and etc. They are usually maintained in isolation and evolve at different rates. Even though all of these artefacts are aimed at facilitating software products developed with the expected quality parameters and fulfilling functional and domain requirements, once produced they often are subjected to different priority levels for their maintenance; some artefacts are hardly maintained and updated as the project progresses whereas certain high priority artefacts, such as the source code, are regularly updated and maintained. This can lead to artefacts rapidly becoming inconsistent with one another and losing their value for development and documentation purposes [1]. Because of this artefact consistency management is a complex challenge in software engineering.

Research on artefact consistency management is an important area of study in software engineering. It is getting popular attracting many researchers who are interested in different aspects of the artefact consistency [2]. Research studies are currently being carried out to tackle software artefact inconsistency problem. A main area of study to solve this problem is the establishment of traceability links among software artefacts. For example, we may wish to indicate that a particular requirement is the reason for the existence of certain design elements in the design specification and such design element can result in a particular type of code snippet thereby creating a faint link between the three artefacts. This may be more straightforward in cases where a generative approach such as Model Driven Development is used to produce one artefact from another, since a mapping is likely to be created and can be used to establish and maintain links in between. However, this single process of generating an artefact from another, which allows clear and easily manageable traceability links, is not always feasible and links may have to be established between existing artefacts and between existing and new artefacts. The default way of doing this is to define the inter-artefact links manually, which is a labour-intensive and potentially error-prone process. Therefore, we establish our main objective of this research as to explore the semi-automatic identification and specification of traceability links among the various software artefacts.

Effective artefact relationship management is a key point for the success of the software process being used for the development. For modelling and storing these artefact relationship links we used graph database approach, which has already been identified as a very effective mechanism for managing unstructured, dynamic relationships and they are especially suited well to store data that already has a graph like structure. These databases natively store graphs with their nodes and edges and offer querying technologies [3]. We use neo4j [4] graph database for relationship modelling, which is one of the most widely used graph databases written in Java; it supports other languages as well. Information on different artefacts, such as requirements, design diagrams and source code, is extracted and stored in a graph database called Neo4j. Graph databases enable data to be stored in a graph structure as nodes and edges. Nodes in this database represent artefact elements such as requirement descriptions and methods in a class. Edges denote relationships among those nodes. This data set is used to analyse the impact of changes to artefacts and to identify the elements that are affected so that changes can be propagated where necessary.

Having a well-engineered visualization system is necessary for a complete toolset of artefact management. We designed a visualizer and a graphical editor over the graph database to support the activities involved in artefact consistency management, such as viewing artefacts in different levels of abstractions, exploring intra and inter artefact relationships, allowing users to traverse the graph for impact analysis and editing the graph to propagate necessary changes to elements and relationships [5], [6].

This paper is arranged into following sections: Section II describes the related work containing the details about