Improving Software Traceability in the Development of Automotive Embedded Systems

A Research Abstract

Salome Maro

Gothenburg University| Chalmers

Abstract Development of embedded software in the automotive domain is a complex task involving the combination of multi-discipline and safety critical requirements. In such an environment, traceability to and from related software development artifacts is demanded by safety standards. It is also needed to facilitate activities such as impact analysis and software maintenance. Despite a lot of research done on software traceability, challenges still exist for complex domains such as the automotive domain due to development being done with multiple companies (suppliers) and integrated at the Original Equipment Manufactures (OEMs) side. This paper describes a PhD research study that is aimed at developing a traceability process, a set of relevant traceability link semantics and a tool. Expected results are that deployment of the process and tool will improve traceability both in terms of link correctness and completeness and that the semantics will support development activities such as impact analysis.

Keywords: Traceability, Requirements Traceability

1 Introduction

Software traceability refers to the ability to link related software artifacts to each other [5]. In the automotive industry, companies deal with development of safety critical embedded systems. It is therefore crucial to be able to verify that safety critical requirements have been well designed, implemented and tested. It is also a requirement from automotive safety standards such as ISO 26262[10] and IEC 61508[3] that companies should be able to demonstrate that this verification is complete. The relationship between artifacts is usually established by traceability links. A traceability link is a specified association between a pair of artifacts, one comprising the source artifact and one comprising the target artifact[5].

1.1 Automotive Embedded Systems Domain

An embedded system is a system whose critical function is not computational but which is controlled with by a computer embedded in it [15]. In the automotive industry most cars manufactured contain software for controlling various functions in the car, ranging from entertainment systems to safety critical functions

such as braking. Software development in this industry is characterized as a complex process having a large number of frequently changing requirements, huge number of variants and configurations, artifacts reuse and specific safety related requirements that need extra attention during development [12], [11]. Due to this complexity, OEMs outsource the development of parts or an entire system to suppliers. This kind of outsourcing means that development artifacts such as requirements are shared between the OEM and suppliers. As a result, it is harder to track how the various artifacts are related to each other and how change in one artifact affects other related artifacts. Therefore, for automotive companies to meet the standards required in terms of traceability, new methods and tools need to be developed to handle this distributed development and product complexity.

2 Problem

Software development activities in the automotive industry lead to the production of a lot of different artifacts such as requirements, design models, code and tests. These artifacts are produced and consumed using various tools, in different teams within an organization and also sometimes across organizations. In industry, it is important to link related artifacts to each other so as to understand how they are related and what should happen if the artifacts evolve. Software traceability has been proposed as a general solution that can be used to establish the relationship between artifacts through the creation of traceability links between related artifacts. To achieve the full potential of traceability usage, the links created need to be correct and complete. Correctness in this case means that the artifacts contained in the traceability link are actually related, given a specific context. Completeness means that all the artifacts that are related are actually connected by a traceability link. In our context, completeness is therefore not a property of one link but a given set of traceability links. The establishment of correct and complete traceability links is not a trivial task because of three major issues that exist. Firstly, there is no clear definition or formalism of what process should be put in place for creation of these traceability links that will seamlessly integrate with the existing software development processes. Secondly, what semantics the traceability links should contain so that they can be useful after they are created and thirdly, what tools can be used to support this. The following sub-sections describe the above three problem areas in detail.

2.1 Traceability Management Process

The process of creating traceability links is time consuming and the accuracy of this step is a major factor in determining if the traceability links created will be useful or not. In existing software development processes such as Agile or the V-model, there is no explicit description of when in the development life cycle, traceability links should be created or updated. This in turn makes the creation and updating of traceability links something that is perceived as an extra activity that adds overhead to the development process. Existing research on traceability is mainly focused on the technical side of the problem by trying

to use techniques such as machine learning [14], [4] and model-driven engineering [13], [7] to develop tools to support traceability. However this research does not suggest a process or how the traceability tools can be integrated in the existing processes. The research outlined here will investigate the existing processes used in automotive industries and come up with a process integrated with semantics and tools that when deployed in industry will lead to the improvement of correctness and completeness of the traceability links created.

2.2 Traceability Links Domain Semantics

Current practice shows that companies only invest in traceability because either their customers or standards demands them and not for their benefits [16] . For traceability links to be useful they need to carry sufficient information that can be used when links are queried. For instance if a project manager wants to know how many requirements have passed their tests the links between the requirements and tests need to have information on whether the test is passed or not. Otherwise the manager will have to navigate to the test manually and see if the requirement passed. Depending on the domain and the uses, the semantics of traceability links can vary, therefore there is no general semantics that can fit all domains. This is what is referred to as "traceability-fit-for-purpose" by the authors in [8] who have defined a road-map for software and systems traceability research. In this research we will focus on determining traceability link semantics that are useful for improving software development tasks for different environments such as product-line and multi-core development environments that exist in the automotive domain.

2.3 Traceability Management Tool

Having a good traceability process is not enough, proper tools need to be put in place so as to support the process. As mentioned previously, there exist research on tools that support the creation and management of traceability links. However, a persistent problem is when the artifacts to be connected reside in various tools and are of various formats (e.g. models, code, word documents), and the traceability links also need to be created and shared by different teams and sometimes different companies. In addition, most tools do not allow for complete traceability throughout the development lifecycle but for only some parts of it, e.g, requirements to code or code to test cases[1], [9]. Research dedicated to supporting traceability in the entire development life cycle also exists but this has lead to solutions that are tool specific [2] and the tools are not configurable and thus cannot be customized for use in other settings. An ideal traceability management tool should be able to integrate easily into the existing development process and be configurable to accept the semantics that are relevant for a different domains. Practical traceability tools e.g. DOORS[1], used in industry also exist. However, such tools only perform well when the traceability links are established between artifacts that reside in the tool. Once you have external

[1] https://www.doorsng.com

artifacts involved other plugins need to be developed to support this. Currently, this leads to inconsistencies in traceability links as artifacts evolve since the tools are not aware of changes happening outside them. Our research will therefore focus on developing a tool that is configurable, can support traceability between arbitrary artifacts and exchange of traceability information between teams and companies.

3 Relevance

Currently many companies spend a lot of effort on traceability in order to be certified by safety standards such as ISO 26262. Coming up with new process and tools to improve traceability will mean that they will save time and therefore costs on their side. It will also mean that they can use traceability for other benefits apart from certification. Moreover, for research, the results of this PhD will give an insight on relevant semantics of traceability links, which processes are suitable, how can they fit in existing processes such as the V-model and also how much benefits traceability actually yields. The research questions to be answered are as follows:

1. RQ1: In the automotive industry, what is the traceability links creation process that can lead to more correct and complete links?
2. RQ2: What are important traceability links semantics that can be used to improve software development activities?
3. RQ3: To what extent can traceability links improve software development activities?
4. RQ3: How can traceability links be maintained and kept consistent when artifacts reside in different tools, teams and organizations?

4 Solution

The PhD is partly sponsored by an ITEA 2 funded project, AMALTHEA4Public[2]. The aim of the project is to develop a common set of tools that can be used throughout the software development life cycle with automotive companies developing embedded systems. The project already has a tool chain called APP4MC[3] that is based on Eclipse. The platform however, lacks any support for traceability and the PhD is part of a work package that is dedicated to investigating the existing traceability problems and providing solutions for the project partners. We expect that our solution will consist of a traceability process that can be integrated into normal software development processes such as the V-model and aligned with safety standards, a set of traceability metamodels that carry semantics that are relevant in different settings, for instance, in product line environments, multi-core development environments or simple embedded development environments. Moreover, our solution will also consist of a configurable tool that will allow companies to easily swap and extend the traceability metamodels depending on their use cases. The tool will also be able to support the evolution

[2] http://amalthea-project.org
[3] http://www.eclipse.org/community/eclipse_newsletter/2015/march/article2.php

and maintenance of traceability links. It will also support the exchange of artifacts and their traceability information between various teams and across organizations. The research will be done in small iterations and in close collaboration with the companies to make sure that what we deliver is not only of value to academia, but also to industry. The research methodology is further described in Section 6. With the possibility of industrial evaluation, we can research the best way for creating traceability links in various contexts, how the traceability links should be used and when and how they should be updated.

5 Novelty

Even though there exist much research on traceability, not much has been done in relation to the embedded automotive industry. As mentioned before, this a complex domain due to the fact that they develop complex safety critical systems and in a distributed environment. For companies in this domain to be ISO 26262 certified, they need traceability. Our original contribution will be a development and integration of the following: first, a conceptual framework comprising of a traceability process for automotive industry and this process will be aligned with the ISO 26262 standard; second a set of meta-models that can be used to support various development environments in this domain, e.g. product lines; third, a prototype that can support the process and semantics and most of all allow for linking between arbitrary elements and exchange of this traceability information between teams and between companies.

6 Research Method

The PhD will be carried out using the action research methodology [6] because solving the described problems requires industrial collaboration. The companies that will be involved are those that are part of the AMALTHEA4Public project. In action research, researchers and practitioners in organizations collaborate in studying a problem, implementing a solution and analyzing the impact of that solution in solving the problem [6]. The obtained results leads to the design and execution of additional iterations that allow progress towards a full solution of the problem.

As depicted in Figure 1, the action research cycles were preceded by a study of the state of the art. This was a literature study that investigated what other researchers have done in the field of traceability and what other tools and solutions have been proposed. The first cycle of the study is done in cooperation with the company rt-labs, a small company developing software for trucks and located in Gothenburg. The cycle began with a state of practice study carried out by sending surveys to all the project partners asking them about the software development activities that they conduct and the artifacts they use and produce. This gave us an idea of the development activities conducted. We also elicited traceability requirements from rt-labs through interviews and focus group discussions with employees at the company. Together with one of the project leaders from the company, we defined a suitable metamodel and implemented a prototype supporting

the metamodel. We will then define an initial process for their traceability use cases and evaluate the process, metamodel and tool at the company by deploying it and getting feedback from actual use of the prototype. This will mark the end of the first iteration of the study. The derived process, metamodel, tool and other lessons learned from this cycle will be carried on as input to the next study cycle. The second, third and further cycles (depending on the need) will be carried out in collaboration with other companies. For now we are sure that the second study will be at Bosch in Germany. The cycles will focus on improving the process, enhancing the metamodel and deriving more metamodels when needed and also adding more features to the traceability management prototype. In each cycle we will also evaluate the results through observing the impact in practice and through controlled experiments with the prototype.

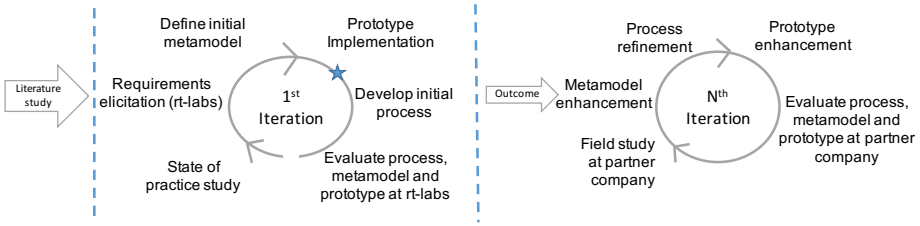

Figure 1. Iterations of the study. The star shows where we currently are in the study.

7 Progress

The PhD started in April 2015 with literature review of the research that already exists in the area of traceability in general and traceability in the embedded automotive domain. We also did a literature survey of existing traceability management tools and what they offer. In May, we sent out a survey to all the project partners to collect information on their workflow. Knowing their workflow helps us in knowing what software development activities they conduct and what artifacts are used and produced in these activities and how these artifacts are related to other artifacts. This information also gave us an insight of which tools are used in creating and consuming these software development artifacts. After the data collection, we conducted two interviews and one focus group with some of the partners on the requirements that they have regarding software traceability.

From the requirements collected, in October we started to implement a prototype for a traceability management tool in collaboration with rt-labs. The aim of implementing this with one project partner is first to get the prototype working on this one company and then expand the evaluation with more partner companies in the project. The next steps for the study will be to conduct a case study at Bosch, because it is one of the companies that are already using the AMALTHEA platform in order to gather more requirements relating to traceability especially when working with the platform. This study will take place in February 2016.

References

1. Antoniol, G., Canfora, G., Casazza, G., De Lucia, A., Merlo, E.: Recovering traceability links between code and documentation. Software Engineering, IEEE Transactions on 28(10), 970–983 (2002)
2. Asuncion, H.U., François, F., Taylor, R.N.: An end-to-end industrial software traceability tool. In: Proceedings of the 6th Joint Meeting of the European Software Engineering Conference and the ACM SIGSOFT Symposium on The Foundations of Software Engineering. pp. 115–124 (2007)
3. Brown, S.: Overview of iec 61508. design of electrical/electronic/programmable electronic safety-related systems. Computing & Control Engineering Journal 11(1), 6–12 (2000)
4. Cleland-Huang, J., Czauderna, A., Gibiec, M., Emenecker, J.: A machine learning approach for tracing regulatory codes to product specific requirements. In: Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering-Volume 1. pp. 155–164. ACM (2010)
5. Coest.org: Glossary | traceability. http://coest.org/index.php/traceability/glossary (2015)
6. Dos Santos, P.S.M., Travassos, G.H., Zelkowitz, M.V.: Action research can swing the balance in experimental software engineering. Advances in Computers 83, 205–276 (2011)
7. Galvão, I., Goknil, A.: Survey of traceability approaches in model-driven engineering. Proceedings - IEEE International Enterprise Distributed Object Computing Workshop, EDOC pp. 313–324 (2007)
8. Gotel, O., Cleland-Huang, J., Hayes, J., Zisman, A., Egyed, A., Grunbacher, P., Antoniol, G.: The quest for ubiquity: A roadmap for software and systems traceability research. In: Requirements Engineering Conference (RE), 2012 20th IEEE International. pp. 71–80 (Sept 2012)
9. Grechanik, M., McKinley, K.S., Perry, D.E.: Recovering and using use-case-diagram-to-source-code traceability links. In: Proceedings of the the 6th joint meeting of the European software engineering conference and the ACM SIGSOFT symposium on The foundations of software engineering. pp. 95–104. ACM (2007)
10. ISO, I.: 26262–road vehicles-functional safety. ISO Standard (2011)
11. Leuser, J.: Challenges for semi-automatic trace recovery in the automotive domain. In: Proceedings of the 2009 ICSE Workshop on Traceability in Emerging Forms of Software Engineering. pp. 31–35. IEEE Computer Society (2009)
12. Pretschner, A., Broy, M., Kruger, I.H., Stauner, T.: Software engineering for automotive systems: A roadmap. In: 2007 Future of Software Engineering. pp. 55–71. IEEE Computer Society (2007)
13. Santiago, I., Jiménez, Á., Vara, J.M., De Castro, V., Bollati, V.a., Marcos, E.: Model-Driven Engineering as a new landscape for traceability management: A systematic literature review. Information and Software Technology 54(12), 1340–1356 (2012)
14. Spanoudakis, G., Garcez, A.S.d., Zisman, A.: Revising rules to capture requirements traceability relations: A machine learning approach. In: SEKE. pp. 570–577 (2003)
15. Wilmshurst, T.: An introduction to the design of small-scale embedded systems. Palgrave (2001)
16. Winkler, S., von Pilgrim, J.: A survey of traceability in requirements engineering and model-driven development. Software & Systems Modeling 9(4), 529–565 (2010), http://link.springer.com/10.1007/s10270-009-0145-0