CHALMERS
UNIVERSITY OF TECHNOLOGY

UNIVERSITY OF GOTHENBURG

How does software traceability affect scope creep in small software projects?

-A Case Study-

Bachelor of Science Thesis in Software Engineering and Management

ALFEDAA SABBAR

Department of Computer Science and Engineering
UNIVERSITY OF GOTHENBURG
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2018

CHALMERS
UNIVERSITY OF TECHNOLOGY

UNIVERSITY OF GOTHENBURG

{The effects of Traceability on Scope Creep}
[How does software traceability affect scope creep in small software projects?]

Supervisor: JAN-PHILIPP STEGHÖFER
Examiner: MIROSLAW OCHODEK

University of Gothenburg
Chalmers University of Technology
Department of Computer Science and Engineering
SE-412 96 Göteborg
Sweden
Telephone + 46 (0)31-772 1000

Department of Computer Science and Engineering
UNIVERSITY OF GOTHENBURG
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2018

CHALMERS
UNIVERSITY OF TECHNOLOGY

UNIVERSITY OF GOTHENBURG

How does software traceability affect scope creep in small software projects?

-A Case Study-

Alfedaa Sabbar
Department of Software Engineering and Management
Email: alfedaa.sabbar@gmail.com
University of Gothenburg
Gothenburg, Sweden

***Abstract*—** This aim of this study is to examine the use of traceability in controlling the scope creep in a software development process in small projects. This study is done on a small project developed by KrisInCorp company to solve their problem of scope creep. Traceability was useful in sorting the requirements and other artifacts in one framework and the tool used produces graphs for the project artifacts and how are they related to each other, this makes it easy to detect the scope creep in the project.

***Keywords*—** Scope creep, software developing, requirements, traceability, Eclipse Capra.

I. INTRODUCTION

In many companies, keeping the scope of project's development under control is a challenging task and is not easy to succeed in. The company KrisInCorp is one of those companies that are facing scope creep in their project development. This study is to determine the causes of this problem and implement a possible solution for it.

A. Problem Statement

The term scope in a software project is defined as the aspects that are covered by the system/developers to be developed [1].

Most of the company's customers are in the United States while the developers are in India, this leads to lack of communication and a better understanding of customer's requirements. Sometimes the customers communicate directly with the project leaders in India and add more requirements than those initially agreed on.

Many clients that the company deals with are not technically experts and have very limited knowledge of software and it's developments. This makes it tricky to understand the exact requirements. This is also one of the reasons why a lot of changes in the requirement list will happen during the development process. When the project is completed, KrisInCorp will be having a bigger number coded files/programming methods than the actual final project, that will lead to more work for the developers, which leads to late hand-over with higher

cost than that which was initially estimated. The company dons not have an accurate cost estimations neither a price policy, because they don't have a good reference to depend on while setting prices. This leads to different estimations for similar requirements or projects.

B. Purpose of Study

In order to improve the current practices in KrisInCorp company, and help them control the scope creep issue, a study is needed to tackle the problem and implement a possible solution to the problem. This would hopefully help the company to control their projects scope and avoid creeping out of it.

What exactly is needed here is to have better control and follow-up on the projects. This could be done by different methods, but for this study we chose traceability. Traceability is the ability to relate artifacts created during developing a system to each other [2]. By making links between the requirements and the coded classes, KrisInCorp can have better control on their projects, because traceability links will make it easy for the company to find the connection between requirements to other artifacts, and if it's connected to the correct artifacts, if not connected to somewhere or if it's connected to wrong artifacts, then it means there is a scope creep with this requirement. This leads to more complications later on in the developing process.

The aim of this study is to introduce traceability in order to help KrisInCorp control scope creep in project development. To achieve this, customer requirements need to be structured in a way which makes it easier for the developers to trace them and use them in making UML diagrams or code classes By listing the requirements depending on their respective types could help to structure them properly.

The purpose of the study is to see if traceability affects scope creep and therefore helping the company to control it. This would hopefully help the company to work time and cost efficiently.

C. Research Questions

- RQ1: How does traceability affect scope creep in small software projects?
  - RQ1.1: What are the causes of scope creep in small software projects?
  - RQ1.2: What are the effects of scope creep on the team and projects?
  - RQ1.3: How can traceability be introduced in a small software company to control scope creep?

II. BACKGROUND

The background of this study is divided into four parts: The company, the Scope Creep and Requirements, Traceability, and Eclipse Capra.

A. The company

The company KrisInCorp is an IT company started in 2011, provides solutions in web, mobile and database technology in the United States of America. KrisInCorp operate on the goal of uniting business with IT in order to facilitate the needs of their customers, thereby leading to increased profitability, transparency and accountability.

KrisInCorp main office is in the United States, while the developers' office of the company is in India with three team leaders and around 25-30 employees, including team leaders, designers, and programmers. The company have had almost 500 customers ranging from small to big companies.

As a small company with two branches, a lot of problems will pop-up in different fields of the company. The major problems happen with the communication between customers in the United States and the developers in India, especially when the customers are not technically educated, which makes the understanding of the requirements not clear enough. This is one of the main reasons behind their scope creep issue.

B. Scope Creep and Requirements

The term scope in a software project is defined as the aspects that are covered by the system/developers to be developed [1]. Creeping in the scope is when there are uncontrolled changes or continuous growth in the project that is not covered by the developers or the changes that were not part of the initial requirements [3].

This study will focus on the scope creep of the requirements. Requirements are the descriptions of the needed attributes, properties or services that the stakeholders requested in order to accomplish the goals of the system [4]. In many projects, the stakeholders will provide different requirements that they would like to have in their project [4]. These requirements will specify the functions of the project and it's objectives, life cycle, operational modes, and interfaces with other projects or systems [5]. There are many types of requirements such as: user requirements, business requirements, functional and non-functional requirements, delivery requirements, process requirements, quality requirements, or/and performance requirements [4]. These requirements are being used by KrisInCorp for their projects. The company uses a mix of these requirements in one list and document in their small projects instead of classifying the requirements into it's types. By not listing the requirements accordingly could cause scope creep since it becomes difficult to make links between the requirements and their respective artifacts.

The aim of using requirement documentation is to have a good understanding of the project that the company/ developers are working on, by that the users will be satisfied while using the complete project with minimum cost and time [5]. Team Leaders list the requirements manually on an excel sheet. Due to the fact that most of the clients are not familiar with the technicality of the project and the requirements makes it difficult for the team leaders to understand what the clients really want.

The company is having a problem with the developing process as there is a lack of reference to the client's requirements. Using traceability would help the company to have a proper reference to the client's requirements, these references will make it more efficient for the company to have more accurate time and cost estimations. Requirements should be written in a readable way in order to manage their development over time by tracing them with other artifacts of the project [6]. Documenting the requirements will make the communication between the developers and stakeholders more effective and understandable [4][6]. A good requirement document is one that is clear, complete, correct, easy to understand, consistent, concise, and feasible [4]. Such document will be difficult to achieve in the beginning since the team leaders make the requirement list manually with no references. But by changing their way of making their requirement list could potentially help them in achieving such document. For instance, making proper references to the requirements, or/and making checklist of the requirements. This could help the team leader to have better understanding and estimation of the requirements that the client is asking for in their projects. Traceability could help the company to have more accurate estimation of time and cost for further projects because it keeps references and sort the requirements with their links to other artifacts.

Using traceability in requirement lists could determine how convenient it is to read and follow up eventual changes to the list [6]. Hopefully, through this study we

will find out if we could control the scope creep issue in this company.

C. Traceability

The term traceability is used in the domain of software engineering to refer to creating links between different software artifacts [7][8]. These links play a major role in identifying the origin of the items and following them as they develop or extend through the developing of a project [9]. It is difficult to imagine a software development without traceability [3].

Specifically the traceability of requirements refers to the ability to define, capture, and follow the traces (links) from requirements to the other elements of the software development environment and vice versa [3][10].

The problem of scope creep is one of the problems that could be solved by using traceability [11]. Traceability could be used to make links between the requirements initially defined and the UML or the code classes. By using traceability links during the development of a project will make it visible if the artifacts are connected to more than one artifact or not connected at all [10]. Traceability links will also help in tracking scope creep issue in the developing process and also what is causing this issues. Traceability links may help control the scope creep and avoid it of happening again in further projects [11]. The team leaders in KrisInCorp could see if there is a sign of creep or not through the presence of incomplete links.

There are several ways in which requirements tracing can be performed in projects. We will discuss two modes (types) of traceability in this study:

- Backward and forward traceability:
  Backward traceability is to trace the requirements to it's own source. And the forward traceability is to trace the requirements to it's components [12]. The forward traceability, traces relations among three software artifacts: requirements, designing, and coding, which are the major artifacts used in any system developing life-cycle [13].

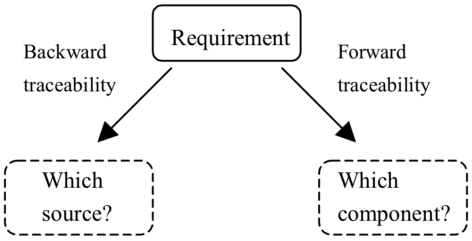

Figure 1 - Backward and forward traceability.

- Inter and extra-requirements traceability:
  Inter-requirements traceability refers to the relationships between requirements it selfs. And extra-requirements traceability refers to the relationships between requirements and other artifacts in the same project [3].

Links between artifacts are those that start from point A (a source artifact) and come to point B (the target artifact) [13]. Both Points A and B could represent requirement, design or code artifacts. Links from point A (such as requirements) could be [13]:

- Requirement-To-Requirement link.
- Requirement-To-Design link.
- Requirement-To-Code link.
- Requirement-To-Others link.

The same type of links could be made for the other artifacts [13].

The importance of traceability has been well studied in the last few years [2]. The main goal of using traceability

is to evaluate how changes to an artifact could affect other parts of the project [13]. Traceability plays an important role in analyzing, change management, testing, and verification and validation in any software development process [2]. It helps the developers to keep the system under control and to manage the changes in the project in order to have better evaluation for future projects [13]. Traceability information can be reused to identify and compare requirements of new projects with those of the existing already developed projects [13].

Finding and tracking artifacts among a software project is not the only purpose of traceability [2]. By using traceability we can confirm if the requirement description was satisfied or not since the links among the artifacts of the projects depend upon the clear understanding of requirements [13]. Also having more trace links will result in the better implementation of a requirement in the project. We can also get an idea if the requirements can be fulfilled by the company or not since some requirements need special tools to design and the company might not have suitable tools [13]. So validating requirements is also a purpose of traceability [13]. Traceability links also help with maintenance, companies that are not using traceability will have to pay higher cost for their maintenance tasks [13]. That is because traceability helps in tracking the source and the extension of each requirement. This will make the maintenance easier such as determining the changes that are needed at a certain point or to find out the location of that particular point. Through the use of traceability, the company could also reuse their project artifacts, resulting in both cost and time effectiveness [13]. Designs, ideas, some codes can be used in different projects and does not need to be created from scratch, by traceability this will be easy to specify and developers could know the project they are building and if it's similar to any other project in the achieve by comparing the requirements list with the traceability track to find what is where. Following links and tracing them makes it easy to locate concrete reusable artifacts [2]. Another example of the importance of traceability could be the ability of ensuring that tasks have been conducted or not, and if yes, if it was conducted correctly as requested or not [13]. Prioritizing requirements and if such task should be implemented or not are decisions that a software engineer or a developer could make by the help of old projects' traceability links [2]. The developer could easily figure out what already exists in the project's archive and what should be developed from the scratch [2].

D. Eclipse Capra

There are a lot of different tools that can make links between the different project artifacts. For this study we chose to go with Eclipse Capra. Eclipse Capra is a traceability tool that is developed by Eclipse Capra development community, which contains some employees of the University of Gothenburg. It is related and tied to a specific technology platform which is Eclipse [14]. Eclipse offers an easier integration with different developed artifacts, that feature makes Capra easy to use and have broader range of use compared to many other tools [14]. Capra also makes it easier to adapt the company's specific traceability strategy; because it allows the definition of the traceability information model to use specific traceability link semantics [14].

Eclipse Capra has some advantages over other tools [14] and is therefore suitable for this study:

- Information Storage:
  Is where the information of the traceability is stored and how easy it can be accessed and consumed by other tools [14]. Traceability information can be used to support activities of software systems such as the change impact analysis, maintenance and evolution [13]. Eclipse Capra stores its traceability information in a specialized model, which allows smaller, co-

located teams, similar to the KrisInCorp team, to treat traces like other development artifacts.

- Level of Integration:
  Capra manages only the links and provides interfaces or adapters for all artifacts. This makes Capra lightweight and easier to integrate with an existing tool chain [14]. For the KrisInCorp problem of scope creep, the type of links we are seeking is the links between the requirements and the code classes, simple links is needed, and Capra will do such purpose. Capra is categorized as a stand-alone traceability management tool [14].

- Integration Context:
  Eclipse Capra is tied to a specific technology platform called Eclipse. If the artifacts of a project are compatible with platform then the integration of the tool with traceability links will be very easy to connect [14]. Projects that KrisInCorp develops are compatible with Eclipse and can be integrated easily.

- Configuration Options:
  The adoption to the concrete environment is easy with Capra because the tool allows to integrate customized adapters to integrate specific artifacts [14]. With Capra, information of traceability can be customized or re-defined as the needs of the project [14]. This may not be needed in our study with KrisInCorp, but it's an advantage that could help in further projects.

- Automation:
  Most of the tools are able to make automated checking of traceability links, such feature will allow creating links to the new added artifacts to the project [14]. This helps projects in further updates or upgrades of projects as well as controlling the scope because these links will show the source of each artifact and the whole path to it's final extension. Capra also supports this feature.

These categories showed the ability of Eclipse Capra, and how it is suitable for creating the links needed in most of the projects that KrisInCorp develops. The tool Capra is also free of charge and is available online for everyone to download and use.

III. RESEARCH METHODOLOGY

This study is a case study, where we have to [15] [16]:

- Design the case study, by defining the objectives and plan the case study.
- Data collection preparation, by defining the protocols of collecting the data needed for the study.
- Collecting evidence, execution with data collection.
- Analyzing the data collected.
- Reporting the results.

The objectives of the case study were defined in the introduction and the background.

A. The plan

This section will focus on introducing a task plan along with a data collection plan. The task plan consists of a number of steps:

1) Interview team-leaders and developers from the company KrisInCorp to know

more about the current scope creep problem.

2) Present the traceability solution to the company and implement it to the development process.

3) Follow up with the new project and collect qualitative data through interviews about the impact of developing the project with the traceability system.

4) Compare (analyze and evaluate) the collected data with the data of old projects that were developed before implementing the traceability system.

5) Summarize the results to answer the research questions.

From the task plan it can be seen which type of data needs to be collected. Initially researching for academic articles, books, and case studies was done to increase our existing knowledge and expand the data that was collected in order to have more information and a better understanding of scope creep problem as well as the uses of traceability.

After collecting data from academic sources we have made interviews with KrisInCorp team leaders and developers to have better understanding of the scope creep problem the company has and how it has affected their development processes. An interview guideline was made (Appendix A). The questions were focusing on understanding the scope creep problem that the company has, how they currently handle traceability management, and to find out where traceability is needed, and possible solutions that could work for them. The interview questions were presented to the team leaders on a Skype meeting. The interviews were also recorded and the data was analyzed. (Will be used/discussed later on result section). More interviews will also be needed later while developing a new project to gather more information about the progress of implementing the traceability on the projects. Finally another interviews will be made with the developers and team leaders in order to distinguish the differences that traceability and Eclipse Capra have made (if any).

The company KrisInCorp have not used any traceability tool before, and nor did the team leaders have any good understanding of the term traceability. It took three online meetings via Skype and TeamViewer to explain the traceability and how it could help the company to observe the scope creep problem and help controlling it. Also, the tool Eclipse Capra and how it is used was presented to the company. Each meeting was with a team leader and one or two developers. The tool was tested on some samples before using it independently in the future. KrisInCorp will develop a new project after implementing the traceability tool Eclipse Capra to trace the requirements of the project with it's components. The traceability will be used by the company in the future as well if they thought it was helpful.

Then it is planned to perform a comparison between projects built before and after implementing the traceability. By collecting information from the developers and team leaders about the outcome of traceability used.

The last part of the plan will summarize the results gathered in the result section and answer the research questions.

Appendixes will contain the interview guides as well as the written answers of the different interviewee before and after the implementation of the traceability.

B. Implementation

To check how useful traceability was in avoiding the scope creep, a small project from the company was chosen. The project was a Job Portal Project and the

technologies used for it were ASP.NET and SqlServer. The first requirement list was made and consisted of 12 points, estimated time for it was 140 hours and had to be finished within 18 days.

The list of requirements was made on Excel sheet and imported to the Eclipse Capra that supported files from Microsoft office and then converted them into artifacts that could be linked later. A team leader was responsible for following the programming and adding completed classes to Capra and connecting them to the requirement list. The developers were doing the same tasks that they were usually doing before implementing the traceability, but the team leader has added that task of implementing the project details to Eclipse Capra and made the links.

Figure 2 - The list of initial requirements list in Capra.

Figure 2 shows the requirements list after importing it from an excel sheet file. The 12 points are the initial requirements that the client asked KrisInCorp to develop. Each of these requirements can be connected separately to any other artifact of the Capra tool. That is why the team leader needed to add the classes to Eclipse and connect it to Capra so the traceability process will take place.

Links are made by dragging the requirement from the "Capra Selection" and the code file from the "Project Explorer" and drop them in the "Trace Creation", and then clicking on "Create Trace" and then choose the type of relation. For this project we did not add any types of relation since were only tracing the requirements to the coded classes. So the only relation we needed was "Related to". A new folder in the "Project Explorer" in Eclipse will be added automatically with the name of "WorkspaceTraceModels". In that folder a "traceModel.xmi" will be created and then the created relation links will be saved in it. In this way the team leader makes the traces between each requirement and it's class. Then, by opening the "traceModel.xmi" and selecting any of it's component (a requirement) a graph will be displayed in the PlantUML section. The graph would also be supported by the Eclipse to work in Capra. This graph will show all the links that are related to the selected item.

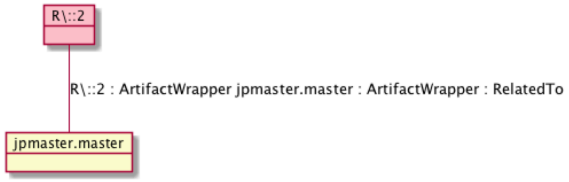

Figure 3 - A trace between a requirement and it's class.

Figure 3 shows how the links will be displayed between the requirements and it's classes. Requirement number 2 is chosen and it's connected to only one class, which is the master web page or template created with ASP.NET and it's called "jpmaster.master"

C. Data Collection

This section will present the data collection plan. Figure 4 shows the data collection plan in correspondence to the task plan steps.

Steps	Task	Subject
1	Identifying the issue.	Research about scope creep, traceability, and implementing new developments to an existing systems.

Steps	Task	Subject
2	Data Collection	By interviewing the developers and team leaders to collect data about the problem of scope creep they face.
3	Capra implementation	Present the traceability to the company and implement it.
4	Tracing	Follow-up and trace the new project and check if the implementation of tracing links are made correctly.
5	Data Collection	Another interviews with the company in order to collect data about the results and the effects of traceability.
6	Evaluate data	Gather and Evaluate Collected Data.
7	Finalize Data	Finalize and conclude collected data.

Figure 4 - Data Collection Plan.

Data about traceability, requirement traceability and scope creep as well as initial interviews have been collected so far and analyzed. Eclipse Capra has been presented to the company KrisInCorp (as mentioned in the background section) with enough details for how to use it and when to use it, and it's now used in tracing an active development progress of a project. The outcome of this application will be discussed in the results.

IV. RESULTS

This section will discuss the results of tracing the project, and the final interviews review.

This project was completed within 190 hours instead of the 140 hours which was initially estimated. The 50 extra hours were due to the addition of 4 new requirements to the project. KrisInCorp took 24 days to complete the whole project. The scope of the project was obviously creeped out. The four extra requirements were added under the developing process. All the initial requirements and the later added ones were implemented to Eclipse Capra, and all of them were traced with links. For the first initialized requirements, 20 different classes were estimated and developed, and for the other add-on requirements, 4 more classes were added. This is shown in the table in figure 5.

Initial estimations			
12 requirement points	20 code classes needed	18 working days	140 working hours
The creep - Added-on			
4 requirement points	4 code classes	6 working days	50 working hours
The project - Total			
16 requirement points	24 code classes	24 working days	190 working hours

Figure 5 - A table showing the creep in the initial scope of the project.

Each item was showing the links that it had with all other artifacts (classes), so team leader could follow and trace each requirement clearly and if it had any issues in it's relations, this made it easy to determine the scope creep. During this time the project faced creep in it's scope, but it was easy for them to detect where the issue was. Interviewee-3 who was the team leader for the developers tested the traceability for this project and said that the links made it easy for them to distinguish where the creep was happening. Interviewee-1 also had a good expression about this experiment, "the tool helped us

organizing the project in a better way, even when the client asks for more tasks. It was easy to add to the tool and keep following them, now we know what was initially agreed upon and what was later added on". Interviewee-2 thought that this was not really helpful, since she was following the project developing daily and didn't find any difference with using traceability or Capra in avoiding scope but she said that this could help in bigger projects that has a lot of requirements. Interviewee-2 also said that these links will helps us reach the classes quickly and provides a reference.

V. DISCUSSION

This section will discuss the answers of the research questions and the threats that we faced during making this study and writing this paper.

A. Research Questions

Answers for the research questions:

- Answer to RQ1: Traceability couldn't avoid scope creep, however it helped in controlling and detecting scope creep on time by making links between different artifacts and relate them to each other, any creep in the scope of the project will obviously be noticed due to the "cut" in the links. When the developer notices the creep, he/she can easily deal with it.
    - Answer to RQ1.1: The causes of scope creep in small projects are many, but in this study communication between the client and the stakeholder was the major problem. Understanding what exactly the client needed was the main cause of not making the scope clear and under control. Another reason we found was the availability of old project's references, that could also be used in developing new project with similar requirements.
    - Answer to RQ1.2: Scope creep affects the team by making them do extra tasks, some of these tasks will not even be used in the final projects. Developers lose time on things that are not used. The company is not only affected by wasting time but also by making extra payments for the unnecessary extra work. Scope creep also effects the company's relation with the client, as the clients may not accept any delays in the submission of the project.
    - Answer to RQ1.3: Traceability was introduced to small companies by presenting it very well to the employees of the company. Presenting traceability, it's benefits, where it could be used, who can use it, presenting some examples, adding some activities that could the employees work on traceability and apply it to the their issues they are facing. Also giving a tutorial of using a specific tool of traceability such as Eclipse Capra was a great idea.

B. Threats to Validity

The main threat to the validity of this paper is not having enough projects to compare, due to the limited time we have for this research, only one project was tested, and this project was also a small project with a list of only 12 requirement points. We were expecting to test at least two projects in order to compare the changes that traceability makes on a developing progress. This project was also the first project that the company used traceability with, which made it difficult for the developers to know what was exactly needed to do and how. Practicing traceability to learn it is easier than actually applying it to actual projects with deadlines, and this also caused delay in getting results.

Another problem was the communication that we had with the company, the company had a difficulty with affording a small project that could be done within the limited time we had. The developers didn't have much free time either for conducting the educational meetings as well as the interviews with the team leaders.

Especially the time difference between the USA and India (where KrisInCorp located) and the local time here in Sweden was a big problem we had to face with the study.

These were the two threats that effected the validity of this study. We have tried to control them by having multiple meetings with the company in order to make traceability and Eclipse Capra as clear as possible, and to understand their experiences about old projects. These meetings also helped us to know how they were handling old projects and tackling any scope creep that occurred in those projects. Due to time zone difference between USA, India, and Sweden, and because of the availability of the employees in either USA or India, we were making ourselves available anytime they were free, even if it was after midnights in the weekends.

C. Further work

More work can be conducted in order to have better results for this study or for further studies about traceability and scope creep.
In our case, implementing traceability in more projects will give more accurate results about the effects of it. Traceability could be implemented to the company future projects and it's effects on scope creep issue could be measured more efficiently.

VI. CONCLUSION

Scope creep is an issue that many companies are facing, and it's difficult for them to avoid it. The solution we presented was "traceability" which we tested with the company to determine it's effects on scope creep issue and it it could help avoiding it or not. After we have tested it and got some positive feedbacks from the company KrisInCorp, the project we tested had also creep in it's scope, so we faced this problem but it was easy to detect where this creep occurs, due to the graph that Capra creates automatically to the related artifacts of any project. It shows that it still needs more test projects in order to confirm the ability of it in controlling scope creep in projects. Traceability helps generally in following the progress of a project, and becomes a reference for the project for updating or upgrading purposes. We chose traceability as a measure to tackle scope creep because it will identify where the creep will happen.

ACKNOWLEDGMENTS

We would like to thank KrisInCorp for their time and effort to complete this study, especially Interviewee-1, interviewee-2 and interviewee-3 for the time they sent in interviews and meetings. Also, we would like to thank our academic supervisors Salome Maro and Jan-Philipp Steghöfer for guiding us and providing us with feedbacks.

REFERENCES

[1] Klaus Pohl and Chris Rupp. "*Requirements Engineering Fundamentals*", 2nd Ed. Library of Congress Cataloging-in-Publication Data, 2015, Ch. 2/8, pp. 8/36.

[2] G. Spanoudakis and A. Zisman. "Software Traceability: a Roadmap", *Software Engineering Group*, pp. 35, 2005.

[3] F. A. C. Pinheiro. "*Requirements Traceability*". Universidade de Brasília. Ch.5, pp 23.

[4] T. Shereni. "*Effective and Efficient Requirement Traceability in the Software Development and Information Technology Industry*". Department of Construction Economics and Management, University of Cape Town, 2014. Ch.1-2, pp 26.

[5] Dr. S. Asghar and M. Umar. "*Requirement Engineering Challenges in Development of Software Applications and Selection of Customer-Off-the-Shelf (COTS) Components*". International Journal of Software Engineering, 2010. pp 18.

[6] B. Nuseibeh and S. Easterbrook. "*Requirements Engineering: a Roadmap*". In Proceedings of the Conference on the Future of Software Engineering, 2000. pp 10.

[7] S. Charalampidou, A. Ampatzoglou, and P. Avgeriou . "*A Mapping Study on Software Artifacts Traceability: Review Protocol*". University of Groningen, 2013. pp 18.

[8] J. Cleland-Huang, S. Rayadurgam, P. Mäder, and W. Schäfer. "*Software And Systems Traceability For Safety-Critical Projects*". Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany, 2015. pp 76-97.

[9] V. Kirova, N. Kirby, D. Kothari, and G. Childress. "EffectiveRequirementsTraceability:
Models, Tools, and Practices", *Bell Labs Technical Journal 12(4),* 2008. pp 143–158.

[10] S. K. Sundaram, J. H. Hayes, A. Dekhtyar, and E. A. Holbrook. "*Software and Systems Traceability*". Springer-Verlag London Limited 2010. pp 313–335.

[11] J. Cleland-Huang, O. Gotel and A. Zisman. "*Software and Systems Traceability*". Springer London Dordrecht Heidelberg New York, 2012. pp 495.

[12] R. Wieringa, "*An Introduction to Requirements Traceability*". Faculty of Mathematics and Computer Science, University of Vrije, Amsterdam, 1995. pp 24.

[13] V. H. Duc, "*Traceability in Agile Software Projects*". Chalmers and University of Gothenburg, Gothenburg, 2013. Ch. 2.1, pp 15.

[14] J.P. Steghöfer, "*Software Traceability Tools: Overview and Categorization*". Chalmers and University of Gothenburg, Gothenburg. pp 6.

[15] P. Runeson, & M. Höst. "*Guidelines for Conducting and Reporting Case Study Research in Software Engineering*". Empirical Software Engineering. Lund University, Sweden, 2009. 14(2), 131.

[16] P. Runeson, & M. Höst. *"Checklists for Software Engineering Case Study Research"*. In Empirical Software Engineering and Measurement,. Lund University, Sweden, 2007. ESEM 2007. First International Symposium on (Pp. 479-481). IEEE.

VII. APPENDIX A

A. Interview guide - Before implementation

1. Purpose of the Interview:

b. To find out their traceability needs and understanding of their scope creep problem.

c. To find out how they currently handle traceability management;

-What is the current process?

-What are the tools being used?

d. To find out what traceability gaps they have (needs) and possible solutions that could work for them.

2. Research questions:

a. What are the causes of the scope creep in small software projects?

b. What are the effects of scope creep to the team and projects?

c. How traceability can be implemented in small software project's company to control scope creep?

d. What are the disadvantages of using traceability to control scope creep?

3. Before the interview:

a. Introduce ourselves

b. Ask to record the interview

4. Background of the interviewee:

a. What is your role in the company? (What do you do?)

b. For how long have you worked in this role?

c. What are other roles that you have had in the past at the company?

d. What development process do you follow? (e.g., V model, Agile, in-between?).

5. Preparation/Planning:

Purpose of traceability

a. How do you follow up the requirements in your developing process. (tools?)

b. When/where do you face the scope creep?

c. What are the effects of the scope creep on your developing process/team?

d. Are you familiar to the term traceability? What is traceability for you?

e. Why do you need traceability links?

- Apart from the current use case, do you have other areas in your development where you need traceability?

Artifacts

a. Which artifacts do you have in your development environment?

- Requirements/Features

- Models

- Wikis

- Code (which programming languages?)

- Tests

- Tickets

*Ask about tools for each artifact if not mentioned with the artifacts.

b. How are these artifacts related to each other? Derived, generated, refined (etc)

c. Do you create traceability links between these files? If yes, how is this done? (e.g, through copying IDs from one file to another?)

d. What is the problem with the way the current links are created?

e. In your opinion, what would be a perfect way to create links? Imagine no tool restriction.

Link Types

a. Are you aware of the term “link type”?

- If yes, what link types do you know of?

- If no, explain what link types are and give examples.

b. In your company, is there a need to have more than one link type or is it enough to have one generic link?

- If yes, why? And which types?

c. What kind of information should traceability links contain?

- Does it make sense to add meta data like, created-by, modified-by/on, reason for existence of T-link and so on?

d. For your role, are high level traceability links better than low level (detailed) traceability links? Why?

Creation/Maintenance

a. When should which traceability links be created? In a particular step in the development?

b. When should traceability links be updated?

- What should trigger the update?

- How do you identify which links need to change?

- Who should update the links?

Output

a. How do you currently store traceability links?/ How do you want the links to be stored?

b. Do you version traceability links?

c. Are there any challenges with respect to the storage and versioning of traceability links?

d. Are there tools that you know of that solve these challenges?

e. How do you want the traceability links to be presented to you? (from customers and the tool).

Uses

a. What do you use the traceability links for?

b. Are there links that are only created and not used?

Requests

a. A project that has stop creep / preferred Java.

b. Teamleader that has developed the studied project as well as the new project.

c. A project files (requirements and traced methods) to make the traceability links before comparing.

6. Conclusion:

a. Summarize what we have learnt in the interview and ask if the interviewee agrees.

b. Do you feel like you have something to add that we missed in the discussion?

c. Can we contact you with follow-up questions for clarification? / ask for another interview meeting later (after implementing Capra).

d. Thank the interviewee for participating.

B. Interview Guide - After implementation

1. Purpose of the Interview:

b. To find out how good was the experience of traceability.

c. To find out how they implemented Eclipse Capra to their project;

-Was it easy?

-how many developers interact with it?

d. To find out the effects of traceability on the problem of scope creep.

2. Before the interview:

a. Ask to record the interview

3. Data gathering:

Results of traceability

a. How do you feel about traceability now after using it?

b. How useful was traceability to you project/ company/team?

c. Have traceability solved the problem of scope creep?

d. what else have been affected by traceability implementation?

e. Are you going to use traceability in further projects?

Eclipse Capra

a. What do you think of Eclipse Capra?

b. How easy was it to learn and use Eclipse Capra?

c. Will you start using Eclipse framework in further projects?

d. Was the graphs of traceability that Capra automated clear to observe the creep in the scope?

e. how many people used Eclipse Capra in KrisInCorp company?

Feedback

a. What do you think that needs to be added to the tool of Capra?

b. With one word, how do you express your feelings about traceability?

4. Conclusion:

a. Summarize what we have learnt in the interview and ask if the interviewee agrees.

b. Do you feel like you have something to add that we missed in the discussion?

c. Can we contact you with follow-up questions for clarification?

d. Thank the interviewee for participating.

VIII. APPENDIX B

A. Interview Answers - Before implementation

1. Background of the interviewee:

a. What is your role in the company?

Interviewee-1: I am a team leader, and sometimes I also develop.
Interviewee-2: Team leader, and CTO, which stands for Chief technology officer.
Interviewee-3: I am developer and also a team leader.

b. For how long have you worked in this role?

Interviewee-1: 2 years.
Interviewee-2: Soon 4 years.
Interviewee-3: I have worked two years as developer and then I also became team leader for the last two years, so 4 years in total.

c. What are other roles that you have had in the past at the company?

Interviewee-1: Nothing, I started as a team leader.
Interviewee-2: I was a developer at first, before I became a team leader.
Interviewee-3: I told you, I started as a developer, and I am still developing while I am also a team leader.

d. What development process do you follow? (e.g., V model, Agile, in-between?).

Interviewee-1: We use Agile now, and sometimes waterfall.
Interviewee-2: We have used waterfall for a while, but now we do use Agile in most of the projects.
Interviewee-3: We use Agile in most, but sometimes we also use waterfall, depends on the project itself.

2. Preparation/Planning:

Purpose of traceability

a. How do you follow up the requirements in your developing process. (tools?)

Interviewee-1: We list the requirements in excel sheet and set a checking list on time line.
Interviewee-2: Manually, using excel sheet and check it when it's done.
Interviewee-3: I do it manually myself, we conduct two scrum meetings a week to check the finished requirements.

b. When/where do you face the scope creep?

Interviewee-1: In the beginning of the project, when we take notes of the requirements, it happens that we misunderstood the client so there it happens.
Interviewee-2: In the stage of writing the requirements, technical gaps between clients and us makes the requirement list changes all the time.
Interviewee-3: When gathering requirements mostly.

c. What are the effects of the scope creep on your developing process/team?

Interviewee-1: We need to re-estimate the project because of the mistakes we do in the initially estimation.
Interviewee-2: we loose focus on the requirements and we waste time because of this.
Interviewee-3: Wasting time and cost.

d. Are you familiar to the term traceability? What is traceability for you?

Interviewee-1: No.
Interviewee-2: No, I don't know.
Interviewee-3: No, I am not.

e. Now you know what is traceability, why do you think you need traceability links?

Interviewee-1: To meet my deadline and have better estimations.
Interviewee-2: I think we need it to trace the quality assurance, testing, and designing, and coding, and requirements.
Interviewee-3: We need it to trace the requirements and have better estimations on coming projects.

Artifacts

a. Which artifacts do you have in your development environment? (Requirements Features, Models, Wikis, Code, Tests, Tickets).

Interviewee-1: We who have requirements, code, tests, and models.
Interviewee-2: Right now we have requirements and coding, we also have testing and the models.
Interviewee-3: Depends on the project, but mostly requirements, code, test, models.

b. How are these artifacts related to each other? Derived, generated, refined (etc)

Interviewee-1: Depends upon the project, but mostly derived.
Interviewee-2: mixed I can say.
Interviewee-3: refined and derived I think.

c. Do you create traceability links between these files? If yes, how is this done? (e.g, through copying IDs from one file to another?)

Interviewee-1: No, we don't do now.
Interviewee-2: No, only manual on a checking list.
Interviewee-3: No. Just the checking list.

d. In your opinion, what would be a perfect way to create links? Imagine no tool restriction.

Interviewee-1: Physically follow-up.
Interviewee-2: Maybe physically, tracing on the checking list.
Interviewee-3: As we do right now, manually by using checking list.

Link Types

a. Are you aware of the term "link type"?

Interviewee-1: No, I don't know what dons it mean.
Interviewee-2: No, what is that?.
Interviewee-3: No, no idea.

b. After you know now what is "link type", in your company, is there a need to have more than one link type or is it enough to have one generic link?

- If yes, why? And which types?

Interviewee-1: I think yes, because we have different artifact and different relations between them.
Interviewee-2: Yes we do, because we have different types of artifacts, and we have to trace them all, so we may need different link types.

Interviewee-3: I am following requirements to other artifacts, I think I need only the "related to" type. But in future, maybe I will need it depends upon the project I will be assigned to.

c. What kind of information should traceability links contain?

Interviewee-1: Requirements number, and the relation to the other artifact.
Interviewee-2: The ID of the requirement, and the ID of the classes.

Interviewee-3: Maybe the requirements number, and the link type, and also the relation to the other artifact and code files.

d. For your role, are high level traceability links better than low level (detailed) traceability links? Why?

Interviewee-1: Depends upon the project, I need detailed information for big projects.
Interviewee-2: For most of projects we do, we need detailed information, this helps us more.
Interviewee-3: No, I think it's enough with a short words.

Creation/Maintenance

a. When should which traceability links be created? In a particular step in the development?

Interviewee-1: The first links should be created immediately after completing the classes.
Interviewee-2: After implementing the classes to link them.
Interviewee-3: When a class is done, link it to the requirements.

b. When should traceability links be updated?

Interviewee-1: Whenever an update occur on a project.
Interviewee-2: Every time we add something to the project.
Interviewee-3: When we add a requirement or a class, we need to make the updates.

Output

a. How do you want the links to be stored?

Interviewee-1: In the computer so we can reach it anytime.
Interviewee-2: As a file on my computer.
Interviewee-3: A file on our computers to get to it quick and anytime. Or maybe save as an image so it will be easy to display and share.

b. How do you want the traceability links to be presented to you? (from customers and the tool).

Interviewee-1: As a graph if possible, but it will do well as a document file.
Interviewee-2: Documented will do fine, but if we can see the graph or drawing, it will be better.
Interviewee-3: Documented as we are used to, but if there is something better like an image, this will be very great.

Uses

a. What do you use the traceability links for?

Interviewee-1: I use it as a checking list, and to know what is related to what.
Interviewee-2: I use those links as a reference for further estimations and to know the relation between the artifacts.
Interviewee-3: To distinguish the creep and as a reference for further estimations.

b. Are there links that are only created and not used?

Interviewee-1: Yes sure, when we have scope creep, it happens.
Interviewee-2: Yes it happens often, this is our scope creep problem.
Interviewee-3: It happens sometimes, but mostly in big projects not on small projects because you know, small projects are easier to control.

B. Interview Answers - After implementation

1. Data gathering:

Results of traceability

a. How do you feel about traceability now after using it?

Interviewee-1: Traceability is really useful, it is a better way for monitoring the project.
Interviewee-2: I liked it, I can see all the relations and what is connected to what.
Interviewee-3: It really helps in finding where the creep is happening. I think it's useful. We can distinguish where the creep is happening and when. Nice!.

b. How useful was traceability to you project/ company/team?

Interviewee-1: I can now monitor the project and use it to know what is done what is not of the requirements as well as finding any problems immidialaty when it happens.
Interviewee-2: The relations that traceability shows helps the company to save time in making updates to the project by quickly finding what needs to be updated.
Interviewee-3: I have used it and it helped me notice that we have faced a creep in the project but the good thing I now know where it happened and how to avoid it next times.

c. Have traceability solved the problem of scope creep?

Interviewee-1: I cannot say yes by only testing it once, but I think it helps finding where is the creep.
Interviewee-2: I think yes, because it showed us what was connected to more than a class and what was not connected at all.
Interviewee-3: yes, sure. I saw the creeps and I also have better reference in order to get back to make updates to the projects.

d. What else have been affected by traceability implementation?

Interviewee-1: This is the first time we used the tool, maybe it took us time to learn it. It affects the times I can see.
Interviewee-2: It helped to have better team work, because we were doing the tasks and implement it to the tool, so each one know what to do next.
Interviewee-3: It took us some days to learn it, but I think with more time we will get more to it and it will keep the process of developing faster and under control.

e. Are you going to use traceability in further projects?

Interviewee-1: Sure yes. We will do in our company.
Interviewee-2: Absolutely.
Interviewee-3: Yes sure, it's so useful.

Eclipse Capra

a. What do you think of Eclipse Capra?

Interviewee-1: The tool is easy to learn and use, and it showed the graphs very perfectly. I liked it.
Interviewee-2: It's a nice tool, we learned it quickly.
Interviewee-3: It's very good tool, made the job perfectly and really helped us.

b. How easy was it to learn and use Eclipse Capra?

Interviewee-1: Not difficult.
Interviewee-2: It took me 3 days to learn using it, I think it was easy.
Interviewee-3: It was easy for me and my collages to learn it. No problems with learning it.

c. Will you start using Eclipse framework in further projects?

Interviewee-1: Maybe for some projects.
Interviewee-2: I think Eclipse would be good for tracing only. We need to change a lot in order to use eclipse, we are now not ready for it.
Interviewee-3: I am using my studio, it's not like eclipse, but I can use eclipse for tracing for sure.

d. Was the graphs of traceability that Capra automated clear to observe the creep in the scope?

Interviewee-1: Yes, it also showed the type of relation, which is really useful.
Interviewee-2: Sometimes it does not show all the graphs, but that problem happens very rare.
Interviewee-3: Yes, it's clear and nice, and useful. I can observe it and follow the links easily on the graphs.

e. How many people used Eclipse Capra in KrisInCorp company?

Interviewee-1: 3 team leaders and 2 developers.
Interviewee-2: We all the 3 team leaders.
Interviewee-3: I am using it, and the other two team leaders.

Feedback

a. With one word, how do you express your feelings about traceability and this experience?

Interviewee-1: Very nice and useful.
Interviewee-2: Wonderful.
Interviewee-3: Interesting.