REQUIREMENTS MANAGEMENT TOOL SUPPORT FOR SOFTWARE ENGINEERING IN COLLABORATION

SAMULI HEINONEN

Heinonen S. (2006) Requirements Management Tool Support for Software Engineering in Collaboration. University of Oulu, Department of Electrical and Information Engineering. Master's Thesis, 91 p.

ABSTRACT

Rigorous competition and tight time-to-market and quality requirements of products are nowadays driving companies to develop products in collaboration. Collaborative development also comes with challenges; communication between parties and product and workflow management during the development becomes more difficult. Especially, requirements management is seen as one of the most critical activities during the development process. In practice, requirements management means the identification of requirements and managing the requirements change. Change management includes requirements traceability between other requirements and between requirements and other artefacts, such as design and source code items and test cases. With the help of traceability, it is possible to see how certain requirement change impacts on the product features and on whole development process.

One possible solution to help the requirements management on collaborative development is the utilization of requirements management tools. In this thesis, it is studied what needs and requirements software industry has for requirements management tools and what kind of software tools are available on the market and how they can support requirements management in collaboration. This study started by analysing and evaluating requirements management tools and their strengths and weaknesses against the collaboration-based criteria. Based on this evaluation, one of the tools (IBM Rational RequisitePro) was chosen for a tryout in a case study, where it was analysed against the criteria with a fictional scenario. In collaborative software development, it is important that different tool vendor's tools can work together and, for example, traceability links between different product artefacts can be shaped. For that reason, RequisitePro and Telelogic Synergy CM tools are integrated between themselves using the Eclipse Framework as a base of integration. This integration demands the development of an extension plug-in between RequisitePro and Eclipse in order that these tools compose a working solution for RM tool support in collaboration.

Based on the case study, we gained clear improvement proposals especially for a plug-in, so that it is easier to continue developing it. The case study also gave experiences and additional information about constructing a collaborative software development environment and thus, helping the development of the tool chain infrastructure. It can be said that processes, practices and agreements are drivers for inter-company collaboration and the role of the tools is to provide discipline for the practices and processes as well as contribute activities for more effective collaboration.

Key words: subcontracting, traceability, tool chain infrastructure.

Heinonen, S. (2006) Vaatimustenhallinnan Työkalutuki Verkottuneessa Ohjelmistokehityksessä. Oulun yliopisto, sähkö- ja tietotekniikan osasto. Diplomityö, 91s.

TIIVISTELMÄ

Kova kilpailu ja tiukat vaatimukset ohjelmistojen kehitysajalle ja laadulle ajavat yritykset kehittämään tuotteita verkottuneesti. Verkottunut tuotekehitys tuo mukanaan myös ongelmia ja haasteita; kommunikaatio eri osapuolten välillä sekä tuotteen- ja kehitystyön hallinta vaikeutuu. Erityisesti vaatimustenhallinta nähdään yhtenä tärkeimmistä aktiviteeteista kehitystyön aikana. Vaatimustenhallinnalla tarkoitetaan käytännössä tuotteelle asetettujen vaatimusten tunnistamista ja erinäisistä syistä johtuvien vaatimusmuutosten hallintaa. Muutosten hallintaan liittyy läheisesti vaatimusten välinen jäljittäminen ja myös jäljittäminen vaatimusten ja suunnittelu- sekä lähdekoodi-tuotosten, sekä testitapausten välillä. Jäljitettävyyden avulla voidaan parhaassa tapauksessa nähdä, millaisia vaikutuksia tietyllä vaatimusmuutoksella on koko tuotekehitysprosessiin.

Yksi ratkaisu helpottaa vaatimustenhallintaa verkottuneessa tuotekehityksessä on vaatimustenhallintatyökalujen käyttäminen. Tässä diplomityössä on selvitetty, millaisia vaatimuksia, tarpeita ja odotuksia teollisuudella on vaatimustenhallinnan työkalutuelle sekä selvitetty, millaisia ohjelmistotyökaluja markkinoilla on saatavilla ja miten ne voisivat tukea vaatimustenhallintaa verkottuneessa tuotekehityksessä. Tämä selvitys aloitettiin analysoimalla valitut työkalut määriteltyjä kriteerejä vastaan. Työkalututkimuksen pohjalta eräs vaatimustenhallinnan työkalu (IBM Rational RequisitePro) valittiin tarkemman perehtymisen kohteeksi ja testattiin sen toimivuutta tapaustutkimuksessa käyttäen apuna fiktionaalista skenaariota. Verkottuneessa tuotekehityksessä on tärkeää, että eri valmistajien työkalut on mahdollista integroida keskenään esimerkiksi jäljitettävyyden parantamiseksi. Tämän vuoksi suoritettiin kokeilu, jossa RequisitePro integroitiin toimimaan Telelogicin Synergy/CM konfiguraationhallinnan työkalun kanssa yhteistyössä käyttäen Eclipse Frameworkia integraation alustana. Tämä integrointi vaati lisäosan toteuttamista RequisitePron ja Eclipsen välille, jolloin muun muassa toiminnallisuutta työkalujen välillä ja jäljitettävyyttä näiden työkalujen tuottamien tuotosten välillä saatiin parannettua.

Tapaustutkimuksen avulla saatiin varsinkin lisäosalle määriteltyä selkeät parannusehdotukset, joiden pohjalta sen kehittämistä on hyvä jatkaa. Tapaustutkimus antoi myös lisäinformaatiota ja kokemusta yritysten välisessä ohjelmistokehityksessä käytettävän järjestelmän pystyttämisestä työkaluketjun jatkokehitystä varten. Verkottuneessa ohjelmistokehityksessä prosessit, käytännöt ja sopimukset ohjaavat toimintaa ja työkalujen rooli on oikeastaan käytäntöjen hyödyntäminen ja tukeminen, jotta toiminta voisi olla mahdollisimman tehokasta.

Avainsanat: alihankinta, jäljitettävyys, työkaluketju.

TABLE OF CONTENTS

PREFACE

This thesis was performed at the VTT Software Engineering Center, Software Development Innovations Team within the Merlin-VTT (E**m**bedded systems **e**ngine**e****r**ing in co**l**labora**t****i****o****n**) project. Merlin-VTT is a part of the Merlin-ITEA project, which is an international research project lasting from July 2004 to June 2007. Merlin aims at improving competitiveness and the product quality in the European electronics industry by providing technological and methodological knowledge to establish competitive collaboration networks between organizations and improving competitive collaboration in Europe by customizing, combining and validating state-of-the-art technologies.

I would like to thank the supervisors of this thesis, Professors Tapio Seppänen and Jukka Riekki, from the University of Oulu. Especially I would like to thank the advisor of this thesis, Ms. Päivi Parviainen, for her competent guidance and valuable comments during the study. In addition, the whole Merlin research team should be acknowledged for their inputs.

Finally, I would like to express my special appreciation to my family for their support and encouragement.

Oulu, January 31, 2006

Samuli Heinonen

ABBREVIATIONS

API	Application Programming Interface is a generic term for any language and format used by one program to help it communicate with another program.
CCB	Change Control Board is the group of people responsible for making decisions to accept or reject proposed changes in software requirements. [9]
COTS	Commercial off-the-shelf, commercially available products that can be purchased and integrated with little or no customization, thus facilitating customer infrastructure expansion and reducing costs. [45]
IDE	Integrated Development Environment is a GUI workbench for developing code, featuring facilities like symbolic debugging, version control, and data-structure browsing.
IPR	Intellectual Property Rights is the general term given to the rights, which encompass such things as patents, licenses, distribution contracts, especially in regard to a new invention, medication, approach. IPRs are the rights of the inventor or assignee to develop and commercialize an invention and license it, usually for a fee, to other manufacturers.
JDBC	Java DataBase Connectivity is a standard for database-independent connectivity between the Java platform and a wide range of databases. The JDBC interface provides a call-level API for SQL-based database access.
QA	Quality Assurance is an integrated system of management activities involving planning, implementation, assessment, reporting, and quality improvement to ensure that a process, item, or service is of the type and quality needed and expected by the client.
RE	Requirements Engineering includes activities, which cover discovering, analysing, documenting and maintaining a set of requirements for a system. [8]
RM	Requirements management manages changes to agreed requirements, relationships between requirements, and dependencies between the requirements document and other documents produced during the systems and software engineering process. [6] It is the process focusing to ensure that customers' requirements will be satisfied in software project.
RM Tool	Requirements Management Tool is understood as a software utility that can be used to help requirements management process by offering certain features such as change management activities, requirements tracing, change impact assessment, etc.

SQL	Standard Query Language is a standard interactive and programming language for getting information from and updating a database.
UNC	Universal Naming Convention is a standard for identifying servers, printers and other resources in a computer network. It simply specifies the name of a particular computer and a particular directory or resource to be accessed. The name of the computer is prefaced with two backslashes, while the directory is prefaced for the traditional one backslash. For example, \\RMserver\dir1.

1. INTRODUCTION

Nowadays software engineering is no longer a one-company activity but rather a team-based activity between several companies. This kind of development is called inter-organizational collaboration. Reasons for collaboration are tight time-to-market requirements, economic reasons, tough competition and complexity of the systems. In order to acquire the required expertise, efficiency and desired lead-time, software products need to be developed globally in collaboration with subcontractors, third party developers and in-house development.

Inter-organizational collaboration helps companies to get the necessary knowledge and labour without their own conditioning process. It also helps companies to reach their goals within certain time and cost limits. In spite of the advantages of collaboration, it also comes with some problems that make the development work more difficult. The software projects quite often fail and usually the reasons are badly defined or misunderstood requirements. Moreover, the requirements changes during the development and the impacts of the changes influence the project success. The changes need to be managed systematically, and especially in case of large projects with high number of requirements, a requirements management tool can help the management process.

This thesis is a part of the Merlin-project (E**m**bedded systems **e**ngineer**r**ing in co**l**laborat**i**o**n**), which tries to solve the appearing problems during the collaborative development. Merlin is an ITEA project lasting from July 2004 to June 2007. It aims at improving the competitiveness and product quality in the European electronics industry by providing technologies enabling faster time-to-market for products and improving the quality of electronics products by increasing the use of systematic development and management practices during product development, distribution and integration. Further, Merlin aims at establishing competitive collaboration networks and improving competitive collaboration in Europe by customizing, combining and validating state-of-the-art technologies.

In practice, the Merlin project aims at developing exploitable solutions for effective high quality embedded systems engineering in collaboration situations, addressing development infrastructures, product quality goals ("the -ilities"), and advanced coding and testing. Moreover, Merlin ensures deployment of these exploitable solutions into industrial practice. This is done through validation of these solutions in different types of embedded software projects, and through experience packaging and exchange. Finally, the Merlin project proposes new standards and de-facto work methods to deploy the identified and newly adapted technologies into a company’s engineering processes.

The Merlin project organization consists of organizations from Finland and the Netherlands. In the beginning of the project, Spain was also included. At this moment, Sweden has also joined the Merlin project. The organizations are companies, universities and research institutes from the afore-mentioned countries. The extensive and valid list of included organizations can be found on the Merlin project web site (see reference [2]).

1.1. The Objective of the Thesis

As mentioned above, software projects quite often fail and the main reasons are badly defined or misunderstood requirements. Moreover, requirements change during the development and the changes influence the project success. Therefore, requirements management is one of the most important issues in software development, especially in collaborative development, and that is why this thesis focuses on requirements management tool support in collaborative environment.

The objective of this thesis is to study what challenges and difficulties appear in collaborative requirements management and how requirements management (RM) tools can help this work. The aim of this thesis is to:

- define the criteria for collaborative RM tools
- study what RM tools are available in the market and how they meet the criteria
- analyse a couple of promising RM tools in detail to help to choose one of them for a case study
- construct a collaborative development environment where the tool chain infrastructure prototype can be tested in a case study
- during the construction, it came out that it is not possible to integrate an RM tool with the other tools without particular tool extension development; therefore this is implemented before carrying out the case study

1.2. The Structure of the Thesis

This document consists of eight chapters. Chapter 1 introduces a short background to the collaboration, describes shortly the Merlin project headlines and gives an overview about the objective and the structure of the thesis.

Chapter 2 focuses on the requirements management and gives a rationalization for requirements management tool utilization in collaboration. It is a so-called background information chapter for the reader.

Chapter 3 describes what the inter-organizational collaboration is and how it is understood in this thesis. This chapter includes discussion about the difficulties in the collaboration and discussion on how requirements management relate to the collaboration. Moreover, a short description of the tool chain infrastructure and of its role in a collaborative environment is given.

Chapter 4 discusses the existing solutions and implementations for collaborative requirements management. The criteria for the requirements management (RM) tools in collaboration are given, tools are analysed against the determined criteria and a summary of the analysis is introduced. In addition, three RM tools are chosen for a detailed analysis, and this chapter finally introduces their features and summarizes them in one table. Rational RequisitePro is chosen from among these tools for a case study, Chapter 6 describes this case study, and experiences of this study are presented.

Chapter 5 reports the development of a prototype, where RequisitePro and Telelogic Synergy are integrated with each other, thereby composing a working solution for a RM tool support in collaboration.

Chapter 7 discusses the research results and importance of the research. It also presents proposals for further studies. Finally, Chapter 8 summarises the study.

2. INTRODUCTION TO REQUIREMENTS MANAGEMENT

This chapter discusses the requirements management in general. A definition for a requirement will be given as well as for requirements management in general and especially what it means in collaborative software engineering. In addition, requirements management activities will be discussed shortly and at the end, there is a short section about using the requirements management tool.

A requirement is something, which is desired or needed. From the software engineering point of view, a requirement is a condition or capability to which a system or software must conform to or a quality that the system must have. Requirements can be divided into two general categories, functional and non-functional requirements.

Functional requirements define the capability and the behaviour of the system. They can be thought of as things that the system does on behalf of the user. Functional requirements are used to express the behaviour of a system by specifying both the input conditions and the output conditions that are expected to result.

Non-functional requirements encompass a wide variety of attributes that do not specifically relate to the system's functionality. These kinds of attributes are, for example, quality attributes such as usability, reliability, performance and supportability. Non-functional requirements are usually as important to the end-user community as are the functional requirements. [1]

Requirements engineering can be split into two main areas of activities, requirements development and requirements management. *Requirements development* concerns activities related to elicitation, analysis, documentation and validation of requirements. It deals principally with the content of the requirements. The purpose of requirements development is to produce and analyze customer, product, and product-component requirements. Requirements development is not within the scope of this thesis. [15]

Requirements management (RM) concerns activities related to controlling and tracking of changes to agreed requirements, relationships between requirements and dependencies between the requirements specifications and between specifications and other project artefacts. Requirements management can be seen as a supportive process, which helps to manage requirements through the product's lifecycle. [5]

2.1. An Overview of Requirements Management

The purpose of requirements management is to manage the requirements of the project's products and product components and to identify inconsistencies between those requirements and the project's plans and work products. Moreover, another purpose is to establish a common understanding between the customer and the software project of the customer's requirements that will be addressed by the software project. [15]

In requirements management, the goal is to meet the stakeholders' conditions and requirements, thus the final result will be the desired product. Activities that are ensuring these objectives are called in general the requirements management. The main task in requirements management is to ensure that the end product comes up to

stakeholders expectations. In the end product, there have to be all the demanded features, and these only.

Figure 1 depicts the main processes of system and software requirements engineering and indicates, how the requirement management is understood as a part of the requirement engineering. RM activities are considered to begin before the actual requirements engineering process phases (RM planning) and to continue during design, implementation, testing and maintenance phases. The figure is adopted from [7].

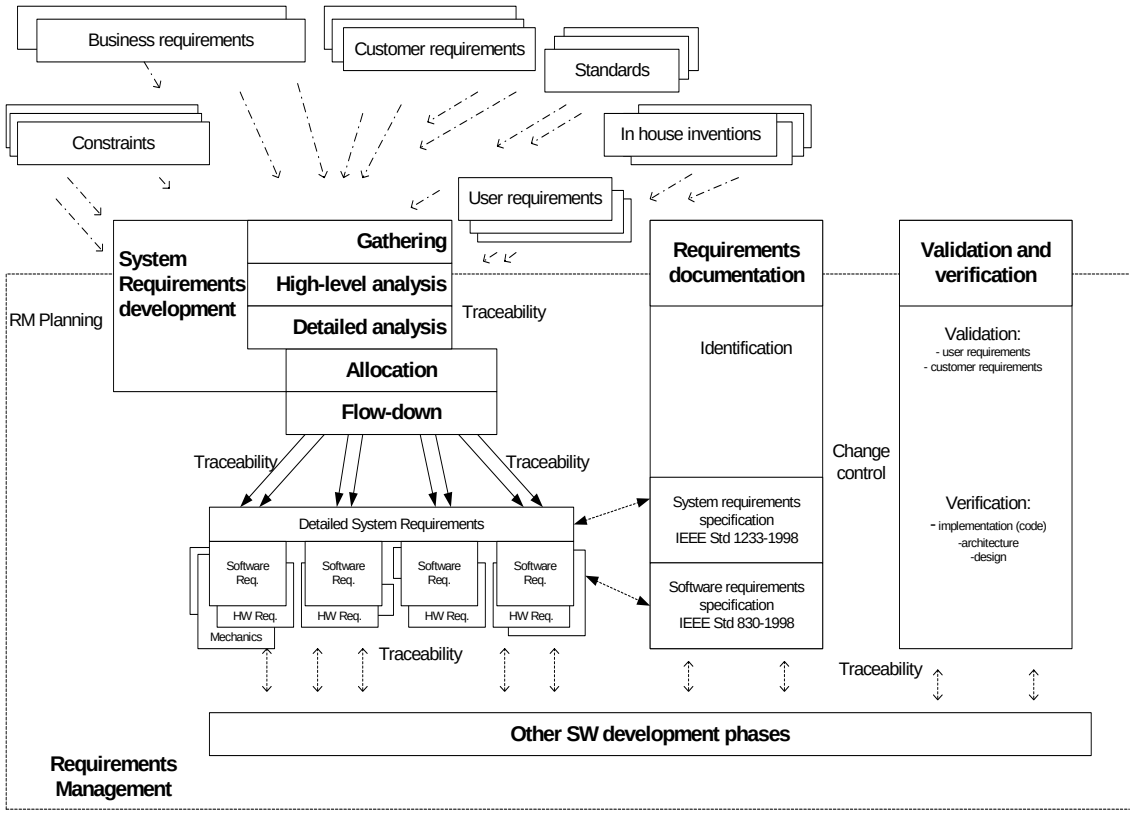


Figure 1. The context of the system and software engineering processes.

Requirements management is the set of activities that concentrate on assuring that the specifications, i.e., requirements, are met to the customers' and other stakeholders' satisfaction. The requirements management manages changes to agreed requirements, relationships between requirements, and dependencies between the requirements document and other documents produced during the systems and software engineering process. [6]

The requirements provide the scope and direction for the projects. Unfortunately, paying attention to the requirements management is not sufficient in practise. If requirements are not managed well, a project can fail or be very costly. On the other hand, a project that meets its requirements is by definition a success. [4]

As mentioned above, requirements management is seen as one of the most critical processes in the collaborative software development. In [6] the authors explain why the requirements management is so important and what happens when the

requirements are wrong. They list a couple of consequences, which arise when the system requirements are not met.

1. The system may be delivered late and it may cost more than originally expected.
2. The customer and end users are not satisfied with the system. They may not use its facilities or may even decide to scrap it altogether.
3. The system may be unreliable in use, with regular system errors and crashes disrupting normal operation.
4. If the system continues in use, the costs of maintaining and evolving the system are usually very high.

Typically, the costs of fixing errors in requirements are much greater than fixing errors made at later stages of the development process. Fixing requirements problems may require rework of the system design, implementation and testing. Under the circumstances, the costs are much higher than fixing a simple programming error. [6]

2.2. RM Activities

According to reference [6], the requirements engineering and design are interlaced activities, that is to say, requirements management continues through almost the whole development lifecycle alongside with other development work.

The requirements management includes a process of managing changes to the system requirements. During the project, requirements can change for a variety of reasons: the changing need of system stakeholders, changes in the environment in which the system is to be installed, changes in the business which plans to install the system, changes in laws and regulations, etc. They say the only constant issue in software development is the changing requirements. These changes have to be managed, so the principal activities in the requirements management are the change control and change impact assessment, better known as *the requirements change management*. [6]

Other activities are *requirements identification* and *requirements tracing*. Requirements cannot be managed effectively without requirements traceability. A requirement is traceable if it is possible to answer to the following questions pertaining to the requirements change: who, why, what, how and when. An essential pre-requisite for requirements management is that every requirement must have some kind of a unique identification. A common method is to identify requirements by numbers. [6]

Under the circumstances, the requirements management can be organised under four main RM activities as follows:

1. Requirements identification
2. Requirements traceability and consistency management
3. Requirements change management
4. Requirements management planning

2.2.1. Requirements Identification

The requirements identification focuses on the assignment of a unique identifier for each requirement, which can be used to unambiguously refer to a requirement during product development and management. Since requirements may evolve during product development, the management of requirement versions is essential. Requirements are usually organised hierarchically, and in consequence of that, parent-child – relations between requirements are possible. This means, for example, if the parent requirement is identified as REQ 2.3, then the child requirements could be identified as REQ 2.3.1 and REQ 2.3.2 and so on. [7]

The requirement attributes can be used to record additional information about the requirements, which helps users to get better management of the complexity of information. Attributes can be included into requirements and they describe information as requirement classification, status, priority, version, relations, etc. Clearly defined attributes can systematise the capture of requirement-related information. [7]

2.2.2. Requirements Traceability and Consistency Management

The requirements traceability is the possibility to describe and follow the life of a requirement in both forward and backward direction. The requirements traceability is used to estimate the impact of changes and to help to understand what the elements where changes would affect are. Moreover, in the later phase of the development, it will enable one to check why a certain feature in the product under development was changed.

The requirements traceability can also cover the relationships with other entities, such as intermediate and final work products, changes in design documentation, test plans, and work tasks. Traceability is in particularly needed in conducting the impact assessment of the requirements changes on the project plans, activities, and work products. [15]

Traceability should cover both the horizontal and vertical relationships. In addition, both of these can be divided into forwards and backwards traceability. The vertical relationship is the traceability between requirements specification, implementation and test, when the forward traceability links requirements to the design plan, product features and test cases, whereas backward traceability links features, tests and design to the requirements. The horizontal relationship refers to traceability between requirements versions, so that forward and backward traceability mean traceability in time-scale. The requirements traceability is seen as a critical part of the requirements change management. [7]

Consistency management is a part of the verification and validation process in software development. Since requirements serve as a starting point for software development, consistency between requirements and following work products should be ensured throughout the development lifecycle. [39]

Traceability would answer, for example, to a question why some feature is designed and implemented, or why some test case was run. Consistency management makes it certain that the produced software will meet its specified requirements. Both requirements traceability and consistency management can be used to ensure that the product really meets the stakeholders' requirements.

2.2.3. Requirements Change Management

The requirements change management is concerned with the processes that are used to manage changes to the system requirements. Information is collected for each proposed change similarly and general estimation is made about the costs and benefits of the proposed change. In other words, impact of changes are managed. [7]

Most of the customers' requirements become clarified during the preliminary study and requirements definition, but it is normal that the requirements change during the project. It is not reasonable to wait until everything about the requirements is known; otherwise, the design phase can never start. Therefore, the change management is a vital part of the requirements management. [7]

The iterative development approach accepts that change will happen. The benefits of continuous feedback need to be realized without losing control of the project scope, quality, cost and timescales. It is important to ensure that the teams working on a particular increment, but at different stages of the lifecycle, are able to determine the correct version of the requirements specification for that increment which they should be working to satisfy. For example, if the user requirements are being modified for increment 2, while the system architects are working on the design for increment 1, the system architects should still be deriving their designs from the user requirements established for increment 1. [48]

It is also important to control the process of change, ensuring that no changes are made without a proper review. This review includes a full impact assessment and it should only be made with appropriate authorization. To effectively analyze the impact of the changes, it is necessary that the source of each requirement is known and the rationale for any change is documented. In order to enable all teams to respond quickly if the change affects their work, all teams should be immediately notified when a requirement, which they are working to satisfy, changes. [15], [48]

Five guidelines to manage changes effectively are introduced in [9] :

1. Recognise that change is inevitable, and plan for it: change procedures for the project should be planned.
2. Baseline the requirements: baseline provides a clear concept for identifying the set of requirements, which are used in a design phase. It provides mechanisms for distinguishing which set of requirements is old and which requirements have changed or evolved after the baselining.
3. Establish a single channel to control change: for example, in large systems a Change Control Board (CCB) who share the responsibility about the approval of the change requests. In small systems, the responsibility can be given, for example, to someone who is an owner of the artefact or a person who has an overall understanding of system requirements.
4. Use a change control system to capture changes: a change control system should be used to capture all requested system related changes. Change requests should be transmitted to CCB for the decision making.
5. Manage change hierarchically: changes to the requirements should be managed in a top-down hierarchical fashion. For example, changes to the specification can cause changes to the features or to design, implementation and test.

2.2.4. Requirements Management Planning

The definition of the requirements management policies is seen as one guideline for RM. These policies include the definition of the goals and procedures for RM, which should be followed in the organisation. According to [8], an organisation can have general policies for RM, but still each project should evaluate their applicability, and select and tailor the related ones for the project. It should be noticed that it is not possible to define the total set of generic policies for an organisation at once. Under the circumstances, they need to be developed incrementally according practical experiences. [8]

The following list introduces the issues for generic policies, which need to be defined:

- Objectives with rationale for the RM process
- Reports which will be produced to increase visibility and activities which will produce these reports
- Standards for the requirements documents and requirements descriptions (templates, structures): this defines the form or template used to record requirements information including the fields to collect the detailed information necessary for the complete specification of the requirement
- Requirement change management policies: responsibilities, procedures and SW support for requirements change management
- Review and validation policies
- Relationship between requirements management and other systems engineering and project planning activities
- Traceability policies: responsibilities, schedules, information types and techniques for requirements traceability
- Criteria when these policies can be ignored. These special practices provide flexibility into the policies, for instance, when there is an urgent need for system changes [8]

2.3. Requirements Traceability Techniques

Many different techniques for requirements traceability can be found in literature, but in this thesis, when requirements traceability is spoken of, it is understood as a direct or indirect link from the requirement to another requirement or to the other product artefacts, such as design or source code items. This kind of traceability technique is called a Simple Links Traceability.

Simple Links Traceability is the simplest form of traceability and it involves establishing a link between two artefacts by using a traceability matrix, hypertext link, graph, or a traceability tool embedded into a requirements management system. A requirement is traced to the artefacts it affects by use of a simple query that returns the set of all directly and indirectly affected artefacts. The primary strengths of this approach are its simplicity and the fact that it is well used and understood in industry, and supported by many requirements management tools. The approach is applicable for many different requirements and artefacts. The major weaknesses of this approach include the long-term difficulty of maintaining large numbers of links, and the static nature of the links that limit the scope of potential automation. [50]

Authors of [50] list, in addition to Simple Links, also other “best of breed” - traceability techniques, such as Semantically Retrieved Links, Executable Links and Tracing NFRs through Design Patterns. These different traceability techniques are usually meant to be used in tracing different types of requirements. Simple Links technique is suitable for functional requirements and Executable Links are suitable for performance related requirements, whereas quality requirements (non-functional requirement) may be linked through design patterns. Moreover, if the best return on investment of traceability process in large projects is desired, it may be worthwhile to use all of these techniques simultaneously.

2.4. An Overview of RM Tools

In this section, the rationalization of requirements management tool utilization is given, and examples of the kind of support they can give for RM process are discussed.

The development of complex systems requires a rigorous approach to requirements capture and tracking. The quality of the end product relies on how well it performs against the customer’s expectations. Customers want to be sure that they get what they asked for. Contractors want to keep track of the current state of the design of the project to ensure that correct design decisions are made with minimal impact on cost and schedule. Requirements management tools are designed to capture the design process and manage the system requirements. [13]

Requirements Management tools are defined to help and ease the software development process. Especially in collaborative software development where development teams are geographically dispersed, the RM tool support is invaluable. Tools can offer many features, such as a general repository, communication capabilities, change control mechanisms and information sharing, which can improve development process and in that way, improve the product quality.

Most of RM tools are based on a database. They may have relatively few records but each of them may include many links, i.e. to documents, text files or other requirements. Commonly used database types are relational database systems and object-oriented database systems.

The RM tools offer several advantages for ensuring requirements traceability. When using RM tools, all requirements and other information are kept in the tool’s database instead of those of different documents and tools, which do not necessarily communicate with each other. The tools provide various methods by which the developer is able to update the links between the unique requirements identifiers. The RM tools integrate requirements management activities as a natural and logical part of the development process. [14]

Most of the existing RM tools have integrated configuration management support, which makes it easy to access the latest available information and makes it possible to find out, for example, when a requirement was changed, who made the change and why it was made. This kind of activity is usually called change management. [14] The change management usually includes a change impact analysis with appropriate notifications, which helps in making conclusions and fixing problems that will be caused by a requirement change.

The tools allow different types of attributes to be associated with each item in the database. The attributes may be used later to organise the data. For example,

different development teams (project managers, software engineers, testing and QA engineers) can create different views to the requirement data. Another example can be a situation, where the project has more than one customer. It should be possible to select only one customer's requirements from the central database before printing them for that particular customer to review. [14]

The RM tools can be configured to give different reports and statistics, for instance, for an inspection meeting or a customer meeting. In addition, some of the tools are able to communicate and integrate with other tools and to store the other tools' data in their database together with ordinary textual data. [14]

Requirements management tools are usually designed for use in quite large projects, so they are not necessarily suitable for use in very small-scale projects.

2.4.1. Requirements Management System

A Requirements Management System (RMS), which can be used to help the RM process, is introduced in [5]. On reflection, it can be said that all requirements management tools are developed based on this kind of RMS on some level. Figure 2 presents almost all the most important functionalities in which the RMS should be supported.

A typical RMS stores its requirements in a repository and provides a number of features to support requirement management activities related to maintenance, evolution, traceability, and change management. In addition, different report generators are typical features in RMS.

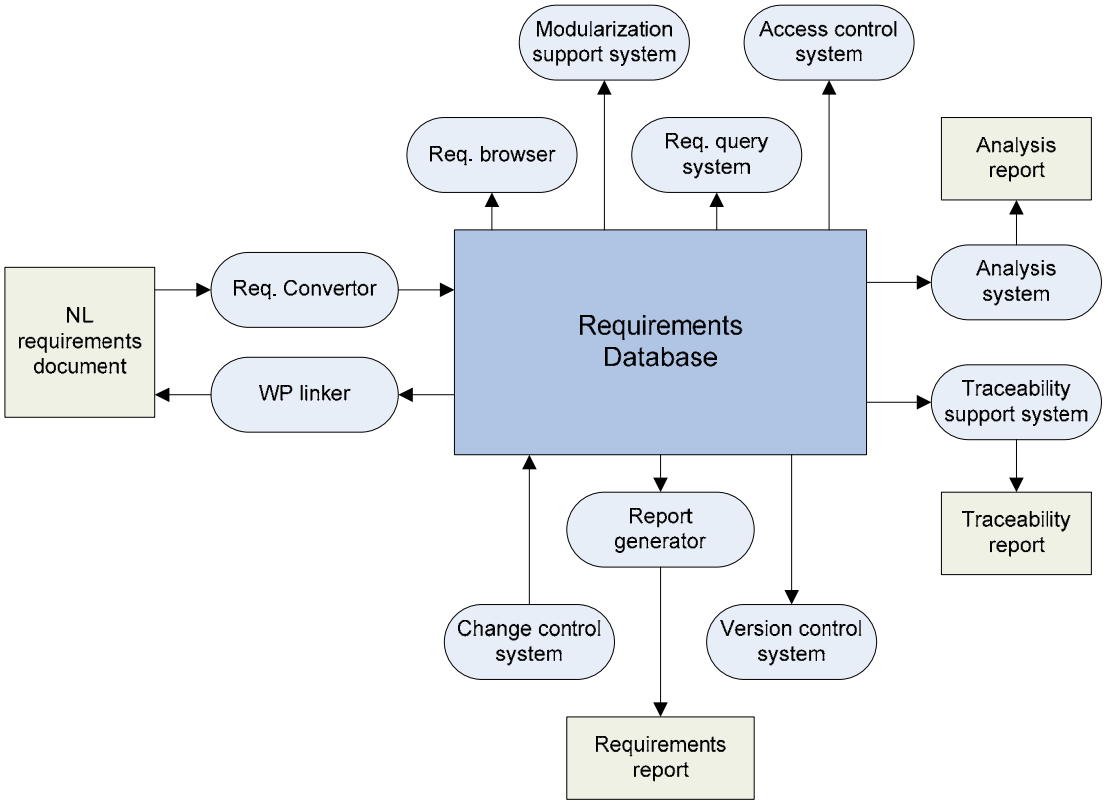


Figure 2. A Requirements Management System.

As figure 2 depicts the typical features of a RMS, the following list clarifies the figure and defines the meanings of the graphical symbols.

- Req. browser; to support navigation in the set of requirements, e.g. view requirements subsets
- Modularisation support system; to support grouping of requirements, e.g. related to a specific quality attribute or piece of functionality
- Req. query system; to support retrieval of specific requirements from the set of requirements or related requirements
- Access control system; to control the access rights of the users. Not all users are allowed to browse or edit the complete set of requirements
- Analysis system; to perform all kinds of analyses on the set of requirements, e.g. impact analysis, status tracking, but also whether a requirement is an orphan or not
- Traceability support system; to support management of links to other system elements and the generation of traceability information
- Version control system; to support management of different versions of single requirements
- Report generator; to support generation of all kinds of different reports related to requirements
- Change control system; to support management of change requests and links to affected requirements
- An interface to external documentation; a typical implementation includes a requirements converter and a word processor (WP) linker, to support conversion of the natural language representation (NL) of the requirements to a database format and back again

The list is not complete. The environment and situation of the project determine the features that should be implemented and on what level of detail. For example, in many situations (local development), it is not necessarily required to implement an access control system, and in the global development the access control system is mandatory. [5]

3. INTER-ORGANIZATIONAL COLLABORATION

Collaboration means that two or more entities work together to create mutual value. Collaboration involves two or more companies, departments or customers that combine their competencies and technologies to create new shared value, at the same time managing their respective costs and risks. The entities can combine in any one of the several different business relationships and for very different periods of time, ranging from some duration needed to exploit a particular innovation or business opportunity, to a much longer-term on-going relationship. [35]

Collaboration in this thesis is understood as an inter-company collaboration, where two or more different organizations are working together. The inter-organizational collaboration encompasses a broad set of actions and practises. This chapter gives an overview of collaboration practises, discussion on reasons for collaboration and a short description of how inter-organizational collaboration is seen in this thesis. It should be noticed that this thesis mainly focuses on the software engineering field.

3.1. Introduction

The development of embedded products has become more demanding. In addition, the lead-time to market in the competitive market has become more important. This is driving companies to global and multi-site embedded systems development. In this equation, global software development is not the target, but it is rather the result of a conscious business-oriented trade-off. Typical for this type of development is the emphatic need of coordination, communication and collaboration.

The continuously growing software products can lead to growing software development teams. It is known that if teams are growing, productivity can become weaker. One solution can be creating several separated teams, which are doing concurrent parallel work. Nowadays those development teams can be dispersed in geography and time all around the world. [10]

According to Merlin Industrial Inventory Summary, the most common motivation for collaboration is to save money. Almost as commonly mentioned reasons are to acquire technology competence or knowledge of a certain market that is not available in house, and there is no interest to invest in company's non-core competence areas. Other reasons for collaboration are to save time, to establish new business opportunities with new partners, flexibility with respect to a number of in-house resources, and availability of in-house resources. In some cases, e.g., COTS developer, pure subcontracting or consulting company, the whole business is based on collaboration. [3]

As can be seen, there are many reasons driving companies towards collaborative development. Although cooperation and collaboration can offer many advantages for companies, it also creates problems and difficulties. The following section describes some collaboration modes and after that, challenges and difficulties in collaboration are discussed.

3.2. Collaboration Modes

Following list of collaboration modes is one viewpoint based on organizational interdependency. The list is also gathered from the Merlin – project Industrial Inventory Summary [3].

Joint R&D (or Partnering)

Joint R&D includes two different ways of action; *Joint ventures* that are organizational units created and controlled by two or more parent companies, and *Joint development agreements* which cover technology and R&D sharing between two or more companies in combination with a joint research or joint development project.

Customer - Supplier Relationships

In customer-supplier relationships, customer is the organization that is buying the software work (and technology and knowledge) from the subcontractor. The work may be based on requirements given, or on modification of the existing COTS or open source code. The customer may also hire workers from a supplier in so called body-shopping. The supplier is the organization that provides the software work to the customer. Three main types of relationships are identified:

- Requirements based subcontracting
- Body-shopping
- MOTS, (COTS), (open source)

Technology Exchange/Licensing

By technology exchange/licensing, the company is granted the right to use a specific patented technology in return for a payment. Companies may also define open interfaces to products that allow any interesting party to create software/services to the product. Types of technology exchange/licensing include:

- COTS
- Open source
- Open architectures

The above categorization is one view to company collaboration; another view is to identify collaboration from the equity or non-equity perspective. Collaboration can also be horizontal or vertical, indicating relative positions of the participating companies on the value-chain. However, in practice, product development is typically a mixture of the above modes, but when analyzing the effects of the collaboration, the above categorization is useful, as it provides different viewpoints to the field.

Figure 3 presents the trend to move from traditional developer-subcontractor relationship towards a multi-partner development. The multi-partner development is more like a network than a traditional subcontracting activity. Moreover, in the collaborative development, there would be more different levels and roles for participant companies than in the traditional development, and these roles are not necessarily fixed.

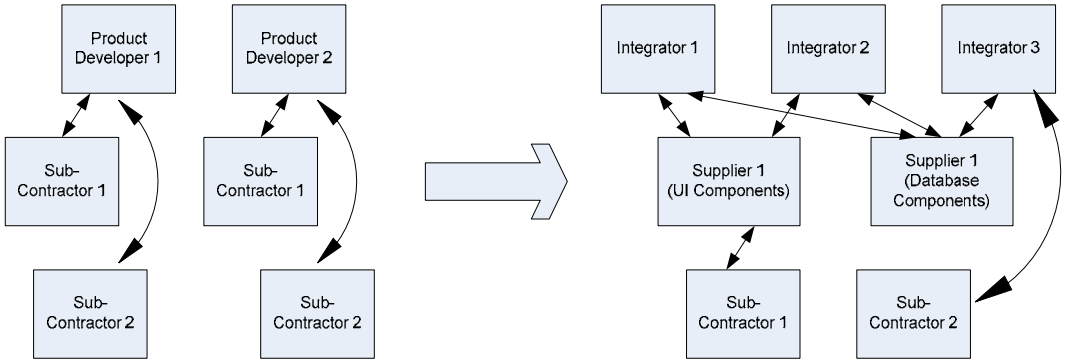


Figure 3. Transition from traditional developer-subcontractor relationship to multi partner development. [2]

3.3. Challenges

The move towards multi-partner collaborative and distributed development comes with many problems and issues to take into account. Collaboration usually means that the project is split across two or more sites and possibly a considerable distance apart. This makes communication, especially informal communication and the quick identification and resolution of problems, more difficult. These difficulties introduced below also hold true in intra-company and distributed multi-site team development. Actually, differences between inter- and intra-organizational collaboration occur in contracts, practices and access levels but the contribution of tools are, however, quite similar in those collaboration modes.

Communication is the basis of all cooperation between humans. It is a process of delivering and changing information between parties. In the multi-site development, the volume of communication usually decreases and it becomes ineffective and insufficient. This is caused by the following reasons: different time zones by decreasing synchronous communication, geographical distance by decreasing face-to-face communication and cultural difference between organizations by causing misunderstanding. Communication is in a central position in almost all collaboration practices and processes, and secondly, it is often mentioned as the biggest problem in the distributed projects. [10], [11]

Coordinating project, which has been split in many sites, is also a demanding assignment. It is difficult to establish an effective relationship with staff who works for other organisations. Hesitant management can leave the team members unclear about what is expected of them. In addition, companies often seem to jump start global inter-organizational projects without first planning how to work together with their partners [11]. This seems to lead to rather problematic situations. To help coordination, it could include, for example, the synchronization of the main milestones. That means it is not always necessary to use the same software development processes but synchronizing the main process milestones between organizations seems to be enough. To increase coordination in a project, it would be advantageous to use frequent deliveries and an iterative development process, for example. It creates transparency, brings real checkpoints, and adds developer motivation. [11]

Cooperation between all parties must be on a high level, if the project wants to be a success. This is true in all teamwork, but especially in collaboration. However, it is not easy to establish a good community spirit between parties, because every staff member in the project has not necessarily even seen each other ever. Arranging for everybody to meet at least somebody from all the other sites seems to be a well-working practice in collaboration. Thus, it would be important to arrange, for example, a kick-off meeting in the beginning of the project, in order to get people familiar with each other. This can help communication, too, because it seems to be much easier to disregard questions or deliveries coming form unknown persons. [11]

Collaboration, as a whole, includes fairly many issues that need to be taken into account, so during the Merlin project the “collaboration handbook” is developed to help setting up and maintain collaboration. It contains such viewpoints that should be taken into account when setting up collaboration with partners. The handbook also includes fields, such as contracts, project management, risks management, change and defect management, quality management and requirements engineering. Every field contains a couple of questions belonging to the main points of the field, and it may give tip links to the potential solutions, which can be e.g. some method or a tool.

During the Merlin project, difficulties in collaboration were studied in literature and industrial experiences and the following findings appeared. In [37], an author collects the problem issues and solution strategies that were regarded as the most important factors in the distributed development, based on the incidence of the issue. According to this study, the most critical factors in no particular order are:

- Physical distance
- Team coordination
- Product planning and integration
- Requirements and specifications
- Lack of trust
- Cultural boundaries in general
- Contracts and Intellectual Property Rights (IPR)
- Architecture related problems

In one questionnaire for Merlin industrial partners, some other issues came up that are critical points when dealing with collaboration. The issues are listed as follows:

- Contracts
- Risk management
- Change management
- Quality management
- Requirements development and management
- Testing and Maintenance

Moreover, the most important solution strategies according to [37] are:

- Efficient informing and communication between parties including practices and infrastructure
- Providing sufficient communication means
- Product architecture is well thought out
- Paying sufficient attention to requirements and specifications
- Competence and experience
- Understanding the partners' culture
- Well managed contracts and IPRs

As can be seen, management issues in software engineering are seen as critical points in collaboration. In addition, issues like lack of trust, contracts, cultural boundaries, etc. need to be taken into account when constructing collaborative requirements management solution. However, requirements engineering is often seen as one of the most critical parts of a software engineering process, and in collaboration, its influence is only growing.

3.4. RM in Collaboration

The requirements management overview has been discussed in section 2.1, but RM differs in collaborative software development compared with normal in-house development. The RM in collaborative environment contains all the same activities than in-house development, but there are many issues that need to be taken into account especially. These noteworthy issues are in many ways the same as mentioned above in the earlier section; communication, cooperation, contracts, working methods, access controls, etc.

In order to make a requirements management process to succeed in the distributed development, it is almost mandatory that there is a RM tool in use. Otherwise, the requirements management needs to be done manually. This means a lot of e-mail transferring, phone calls, document handling and other management practices such as change and defect management between the dispersed teams. All of this is almost impossible to do effectively without appropriate tools. Of course, this is true particularly in large projects where there are a number of requirements and the teams are large. On the other hand, if the project is quite small, an RM tool is not necessarily suitable for use, because it can make the development work a little cumbersome. However, even in small projects, if distributed, centralised RM tooling can significantly shorten the time for the RM and specially the change management.

The first step in collaborative requirements process is to ensure that all stakeholders understand what the project goals are and why. Therefore, the organizations must create a process where the stakeholders are invited to help to define the requirements and to agree on a baseline. As the project progresses, the organizations can then trace the results back to the original goals and ensure they have been met and that any changes have been justified. In order to collaboratively create and manage a defined set of requirements, the requirements management tools should allow multiple users to view and edit requirements documents or database. The users also need to establish traceability links between requirements, design documents and test documents. This ensures that the requirements have been met, even as they change. Moreover, it is important to maintain the history of the requirements changes. [18]

When physically distributed teams are working together on a single project, the requirements database and its consistency management have to be supported [16]. In addition, a central repository for all project data is one proposed decision. In a situation, where a central repository is in use, the requirements traceability can be implemented easier, because all the project data can be stored in the same place. This helps at least tracing and maintaining the issues.

The RM tool must provide different views with respect to roles and process activities, where only a subset of the existing information is presented to the user.

For example, a project manager is interested in different information details than a QA expert, a requirements specification document is a particular representation of a view on the existing information related to the requirements. [16]

Regarding the security aspects, the access to particular information must be forbidden for some users. E.g., the QA staff should only have read access to requirements, whereas the cost information is exclusively readable for the project managers. Moreover, the staff from certain organizations' may have different access rights to the database than another, in order to meet legal requirements. A prerequisite for definitions of distinct views and access rights are the defined roles and user profiles. The RM tools usually support several forms of views and fine-grained definition of access rights. [16]

In [11] they say that inter-organizational software projects are often faced with uncertainty regarding the requirements and implementation techniques. This is due to the need to involve subcontractors or partners long before these things are resolved. Parties cannot receive a clear requirements specification at the beginning of the project. In such cases, close cooperation and communication between parties is required during the whole project life cycle, as the project both builds a product and, at the same time, tries to understand what to build.

Of course, the requirements management in collaboration requires much more than supporting tools. It can be said that working practices, standards, agreements and processes are the drivers and tools are the enablers.

3.5. Tool Chain Infrastructure

A Tool Chain Infrastructure means software tool integration, which supports some interactions or activities during the software development lifecycle by offering several services in the same development environment. A tool chain can occur inside a company or it can concern collaboration between several companies.

In the inter-organizational collaboration, the ideal situation is, that all software development tools are integrated with each other, composing the Tool Chain Infrastructure, where information can be transferred from one development phase to another and also between different tools in the same phase between organizations. Figure 4 clarifies the idea of the tool chain infrastructure in a simplified environment.

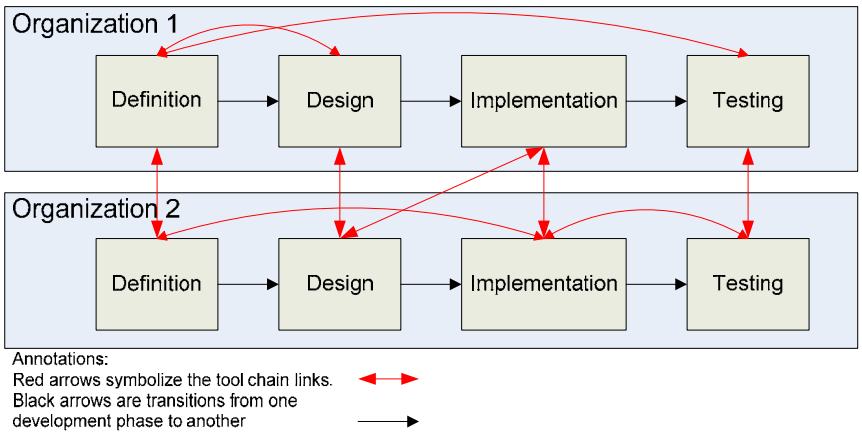


Figure 4. The Tool Chain Infrastructure.

A tool chain must not necessarily cover the whole development chain, but it can also be shorter. For example, integration between the RM tool and the Testing tool is one possibility. Integrations between tools are necessary in order to gain capable requirements traceability and a change management system.

The tool chain infrastructure is comprised of integrations between software tools. Information sharing and transferring usually implement integrations between tools, but there can also be functional integration between the tools. Figure 5 presents a couple of examples for possible integrations between requirements management tools and other development tools.

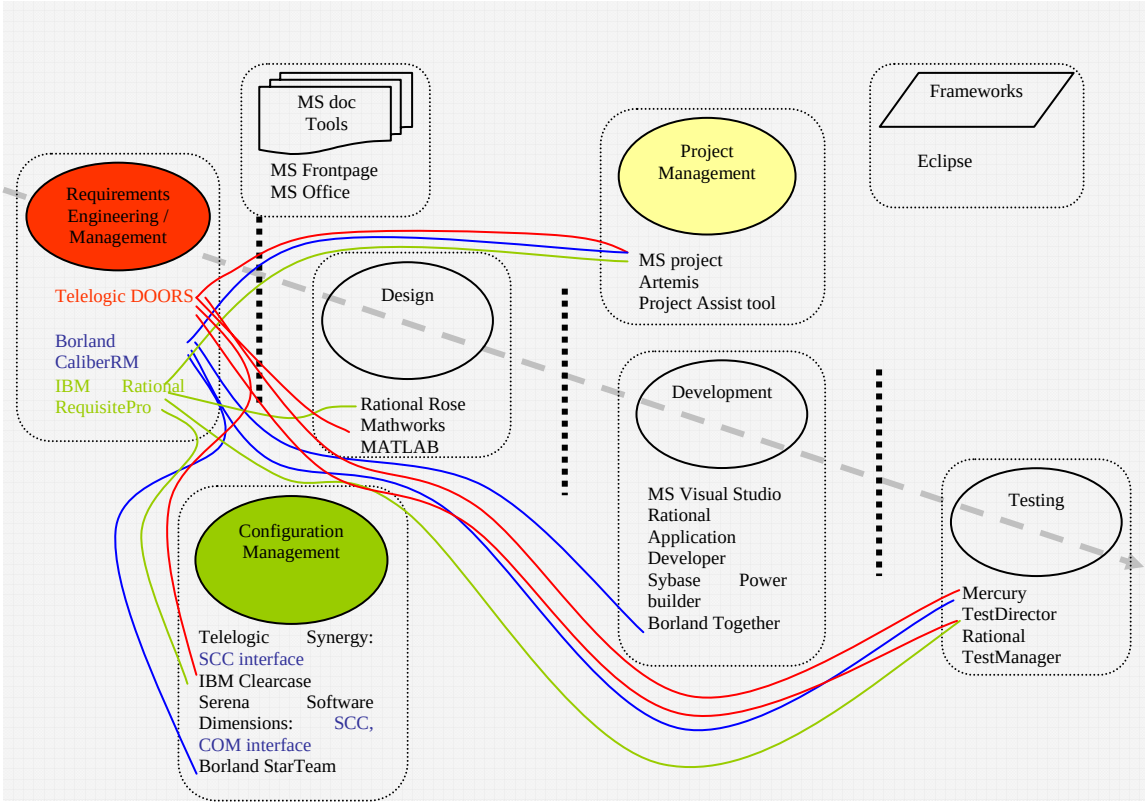


Figure 5. Tool integrations from RM viewpoint.

Let us examine a couple of integrations from RM tools to other tools and how those integrations are implemented in reality. For example, integration between Telelogic DOORS and Microsoft Project enables the project managers to create project plans from requirements document, communicate the schedule dates to the project team, and assess the impact of the changes of the project schedule or requirements. The project plan can also be transferred from MS Project into DOORS, when links can be created between activities and requirements. Creating a preliminary project plan from requirements helps to ensure that it meets the customer's need. The integration increases the deadline visibility to the project team and facilitates the assessment of the change impact. [40]

Further, also RequisitePro can be integrated with MS Project. This integration is done by creating project tasks from the requirements and establishing traceability relationships from those requirements to the tasks. Of course, it is possible to

establish traceability relationships from the existing project plan to the requirements or vice versa. When traceability relationships are created, it is possible to analyze the requirements change impacts on the project schedule, if a certain requirement changes. In addition, the project progress reporting becomes easier with this integration. When integrating RequisitePro with MS Project, MS Project commands are displayed on the RequisitePro Tools menu. To make this integration possible, the user must have both tools installed in the system. [44]

Another example is integration between Rational RequisitePro and Rational TestManager. By using TestManager, the user can establish requirements from RequisitePro as an input to a testing project. In that case, TestManager uses the RequisitePro's project file as an input source. Once a source has been established and test cases are associated with requirements, TestManager keeps the track of each association. [41]

In addition, Borland CaliberRM offers integration with Visual Studio .NET. CaliberRM requirements data is displayed within the VS .NET IDE. The user can modify requirements by selecting a requirement in the VS .NET IDE and launching CaliberRM which automatically navigates to the selected requirement. The requirements may be dragged into the source code as a formatted comment block. The comment syntax and style can be chosen and which elements of a requirement make up the composition of the comment. It is also possible to scan through all the source codes by the means that all outdated requirements are identified and automatically updated. In that case, the integration is implemented as a VS .NET add-in. [42]

The integration does not need to be just a physical connection between separate tools, but it can also be a tool support for some design method. The tool chain can occur inside a company or it can concern the collaboration between several organisations. One possibility to support collaborative development environment is to use Application Service Provider (ASP). An Application Service Provider is a company that offers individuals or enterprises access over the Internet to application programs and related services that would otherwise have to be located in their own personal or company computers. The ASP concept offers opportunity for smaller companies to lower their information systems related costs. Therefore, it enables the use of similar tools inside the collaboration by renting them instead of buying. [19]

The benefit of the tool integration is faster development work through a fluent data transfer between different tools inside and between organisations. The tool chain gives possibilities for more coherent and comprehensive data transfer through the steps of software life cycle by minimising the changes between several development environments, misspellings, double actions, forgetting, etc.

In order to be a well working system, the tool chain should perform reliably and quickly. Security issues are essential, especially when collaboration is in question, and of course, the usability should not be too challenging. Before integrating, it should be noticed that the integration has a clear purpose and sufficiently high value and the increased complexity can be managed. [19]

4. RM TOOL SURVEY

This chapter discusses the existing solutions and implementations for the collaborative requirements management. The tool survey started by defining the criteria for the requirements management tools in collaboration based on information found in literature and in earlier projects. The criteria were updated and improved based on the Merlin workshop discussions. After that, a broad list of RM tools which appear to support collaborative software development, is collected, based on the information found on the vendors' web pages and in literature.

4.1. Criteria for RM Tools

As mentioned before, requirements and criteria for RM tools were gathered at first in literature. In VTT, there have been a couple of previous projects, where requirements management was within the scope, such as the MOOSE project [38]. In addition, a couple of research papers have been published where requirements for tools are listed, but they are not based on the collaborative viewpoint. Therefore, requirements were also defined based on company interviews and comments in workshops with the Merlin partners.

Developing the criteria for the collaborative RM tools started with the literature study. Requirements for the RM tools are presented in a couple of research articles and earlier projects. For example, sixteen requirements for the inter-organizational IT support (IOIS) for collaborative product development are introduced in [12]. These requirements should be seen as guidelines for collaborative tools to improve communication with other parties. Further, [20] presents a set of requirements for RM tools in the area of automotive and aircraft industry. In addition, in [21] authors have defined a requirements list for the RM tools used in software engineering in collaboration, and a detailed list of characteristics of requirements management tools is introduced in [8].

By integrating the lists of requirements and adding the collaborative point of view, preliminary criteria for the RM tools were defined. After that, a workshop was arranged with industrial and academic partners of the Merlin project where the preliminary criteria for the tool evaluation was improved and updated. Pertaining to this workshop, pre-inquiry was sent to the Merlin industrial partners about tools they are using for project management, requirements management and configuration management. The inquiry also contained a section where the respondents could express their interest for tools that need further consideration in the Merlin project and a section where they are able to define the good and the improvement issues for the tools.

The criteria are divided into two parts, general criteria and specific criteria. The general criteria fit all software tools that support collaborative software engineering, whereas specific criteria include requirements that are specific for the requirements management tools. After the criteria tables, there is a detailed definition and rationalization for each single criterion.

4.1.1. General Criteria for Collaborative Tools

During the RM tool survey, other software tools that can be used in collaborative software development were also studied pertaining to the Merlin project task. These other tool categories were project management (PM) tools and configuration and product data management (CM/PDM) tools. The common criteria for all these tool classes are presented in Table 1. These criteria are important for tools that are meant to be used in collaborative environment.

Table 1. General criteria for collaborative tools

Criteria	Description
Usability, Simplicity and Customization	The tool should be easy to use. Not too much training and administration needed. The tool should not create additional tasks and deployment should not require extensive customization.
Multi-platform support	The tool should support multiple operating systems. (Windows, Linux, Unix…)
Tool integration	Integration to the other tools should be supported.
Web access	The tool should have a web interface that makes it unnecessary to install a client application for occasional users.
Access control	The tool must have tight access control whereby each participant has appropriate access to the data. (Role-based, project-based and task-based access control.)
Information sharing	The tool should support information sharing whereby all participants can be up-to-date. (E-mail notifications, news groups, discussion forums, etc.)
Simultaneous use	It should be possible for many users to work on the same data at the same time securely.

Usability is an obvious need for a tool supporting collaborative way of working. In order for companies to take tools in use, the tool should not create additional tasks and complicate the development work. In addition, **simplicity** (e.g. training and administration) and ability to operate without extensive tool **customization** are important factors, especially for small companies.

Multi-platform support means that the tool can be run under different operation systems, such as MS Windows, UNIX and Linux. The multi-platform support is important and useful in collaborative environments, because companies do not necessarily have the same data systems.

Tool integration is needed, for example, to simplify information transfer between systems, to provide the chain of applications to support, for instance, certain development methods and to launch the functionality of a tool from another tool. Tool integration makes it possible to create tool chains that physically link the systems and methods for the product development in collaboration.

Web access is needed in situations where a certain part of personnel are working in external location and need to have access to the tool database once in a while. For example, a salesperson who is with the customer can use the web interface to check which requirements and features are to be implemented in the next product version.

Access control is important in collaborative development environment since, for instance, persons from external organisation should not see all proprietary information in the company's data systems. Moreover, it is not necessary for developers to see the project budget, for example, and a Quality Assurance (QA)

person can only read requirements, editing is not possible for them. The tool should support restricting a particular user group's access to certain information and, in general, control accessing to the tool by passwords and data protections.

Information sharing is very important in collaboration, so that all parties keep up-to-date and useless labour can be avoided. Communication is seen as one of the biggest problems in collaboration, thus supporting it is worthwhile. Internal deliveries of products or parts between geographically dispersed teams are done with the help of different tools. Employees have access to the same data (as per on need-to-know basis). Moreover, if a requirement changes, the tool can send email notifications for appropriate persons noted as responsible for the requirement. The notification can be sent to the groups of users for certain requirement changes. Threaded discussions could also help information sharing between distributed teams.

Simultaneous use is an obvious need for collaborative systems. The need for simultaneous use is, for example, the situation where requirements developers are geographically in different places, and they need to define the same set of requirements at the same time in co-operation.

4.1.2. Specific Criteria for RM Tools

The specific criteria are defined for every tool category during the tool survey. The following table introduces the specific criteria for the RM tools.

Table 2. Specific criteria for RM tools

Criteria	Description
Requirements identification	The tool must support the identification of requirements. For example, ID – number for each individual requirement is mandatory.
Requirements classifying and viewing	The tool must be able to classify requirements into logical user-defined groups. Moreover, the tool must support various views of the same data.
Formats	The tool must be able to specify requirements with textual, graphical, and model based description.
Change management	The tool must offer a possibility of handling formal change requests. All changes to the requirements must be tracked and kept in the database.
Traceability	The tool must enable traceability through links between requirements. Moreover, the tool should support traceability from requirements to features, product versions, customers and components.
Document importing	The tool should be able to import existing requirements specification documents while retaining the structure of the content. In addition, semi-automatic requirements importing from documents should be supported.
Document generation	The tool must be able to generate official and internal documents. The tool uses predefined document definitions to generate documents with current data from the database.
Tailoring and Extensibility	The tool must be adaptable and extensible to the needs of the organization or project.

Requirements identification means the ability to identify every single requirement so that distinguishing them from each other is easy. This can be done

with requirement identification numbers and with the help of requirements attributes. In addition, the tool must support requirements prioritization, because some requirements are more important than others are, and features that are more important have to be implemented first.

Requirements classifying and viewing is the ability to classify requirements into logical user-defined groups, thereby offering different views of the same data to the different users. A view offers the possibility to view and change a freely defined collection of parts of the data of several projects in a freely configurable representation.

Formats is understood here as an ability to specify requirements with textual, graphical, and model based descriptions. Standard system modelling techniques and notation should be supported. (UML, Use cases, etc.)

Change management is the most important feature in the requirement management tool. The tool must provide a possibility of tracking all changes to the requirements and keep them in the database. The history of the requirements changes (who, what, when, where, why, how) needs to be registered. Change management usually includes many actions, such as e-mail notifications and impact analysis. Change management is especially important in the collaborative development, in order to keep the development process going correctly.

Traceability means traceability through links between requirements. It is also the ability to define traceable associations between requirements. Moreover, it can be traceability from requirements to features, product versions, components and design documents. Traceability is an important feature in change management activities. It is also important in later project phases, when assuring the consistency of requirements and other product artefacts.

Document importing makes the handling of the requirements document easier. Users can create requirements documents with a customer without access to the RM tool. Later, the requirements document can be imported to the requirements database with the help of the tool. Requirements can be transferred from document to the database semi-automatically, one by one, or even the whole document can be stored in the database.

Document generation means that the desired requirements are gathered from the database to the requirements document. It is not practical to print the whole contents of the database to the document, but only the appropriate requirements. For example, a certain customer is only interested in the requirements that relate to him. The documents could be ad hoc reports, predefined reports, or comply with standard industrial templates.

Tailoring and extensibility is practical when the company has many projects of different sizes, and many different tools are used with the RM tool. The tool must be easily adaptable and extensible to the needs of the organization or project.

4.2. Summary of Preliminary Survey

Tools introduced in this chapter are taken into account based on how the tool vendors promise to support collaboration. In accordance with their web sites and white papers, potential support for the collaborative development is estimated. The advantage of this method is that functions and features that tools provide for collaboration can be directly found. Another approach has been to set conditions and

requirements to the appropriate tools first, and then analyze how they meet those requirements.

A set of requirements management tools was chosen for the study. The number of RM tools is huge, but this study did not include all of these. The selection criteria for tools included were that they are well known and broadly used in industry. In addition, information on these tools is quite well available in the literature and in the Internet. The following list introduces the tools chosen for the study.

- Borland CaliberRM
- Sophist Group CARE 3.2
- Steeltrace Catalyze Enterprise
- ViewSet PACE
- Insoft Prosareq
- IBM Rational RequisitePro
- Igatech Systems RDT
- RBC RMTrack
- Serena RTM
- SpeeDEV RM
- Telelogic DOORS
- Teledyne Brown XTie-RT

All of these tools are analysed against the defined criteria and Table 3 illustrates the summary of the results. It should be noted, that the first criterion in the table (Usability, Simplicity and Customization) is left blank, because evaluating usability should demand a test use with every tool and it was not possible, because of lack of time and resources.

Table 3. Summary of the preliminary study

General criteria	CaliberRM	CARE 3.2	Catalyze Enterprise	PACE	Prosareq	RequisitePro	RDT	RMTrak	Serena RTM	SpeeDEV RM	Telelogic DOORS	XTie – RT
Usability, Simplicity and Customization												
Multi-platform support	-	+	-	+	+	-	-	-	-	-	+	-
Tool integration	+	+	+	+	+	+	?	+	+	+	+	+
Web access	+	+	+	+	+	+	+	-	+	+	+	-
Access control	+	+	+	+	+	+	+	+	+	+	+	+
Information sharing	+	?	+	+	?	+	?	-	+	+	+	?
Simultaneous use	+	+	+	+	+	+	+	+	+	+	+	+

Specific criteria	CaliberRM	CARE 3.2	Catalyze Enterprise	PACE	Prosareq	RequisitePro	RDT	RMTrak	Serena RTM	SpeeDEV RM	Telelogic DOORS	XTie – RT
Requirements identification	+	+	+	+	+	+	+	+	+	+	+	+
Req. classifying and viewing	+	+	+	+	+	+	+	+	+	+	+	+
Formats	+	+	+	+	**?**	+	-	+	+	+	+	+
Change management	+	+	+	+	+	+	+	+	+	+	+	+
Traceability	+	+	+	+	+	+	+	+	+	+	+	+
Document importing	+	+	+	+	**?**	+	+	+	+	+	+	+
Document generation	+	+	+	+	+	+	+	+	+	+	+	+
Tailoring and Extensibility	+	**?**	**?**	+	**?**	**?**	-	**?**	+	**?**	+	+

Classification:
+ = the tool supports this criterion
? = evidence about support can not be found or it is unclear
- = the tool does not support this criterion

As can be seen in the Table 3, traditional and general requirements management functions, such as requirements classifying and viewing, change management, traceability and document generation are well supported by every tool. Of general criteria, tool integration, access control and simultaneous use are also well-supported features by every tool. On the other hand, support for multi-platforms is weak in general.

Usability, Simplicity and Customization are considered criteria that are difficult to evaluate without test use. Due to lack of time, these criteria were not evaluated in this case. It should be noticed that these results are only indicative, because assessment data is gathered from vendors' web pages and other available evaluation report and white papers. However, results give some kind of a review to the RM tool support for collaborative requirements management nowadays.

4.3. Summary of Detailed Analysis

The aim of the detailed analysis is to analyse three RM tools exactly, and thus help to choose one of the tools for the case study. The following three tools are chosen for a detailed analysis, because these tools meet the criteria best: Borland CaliberRM, Rational RequisitePro and Telelogic DOORS. The selection is also done based on the Merlin industrial partners and VTT interest.

The tool mapping presented in the earlier chapter was a preliminary mapping between the tool functionality and the criteria. The preliminary mapping was based on information gathered from vendors' web pages and evaluation reports. The detailed analysis was also done based on web pages, evaluation reports and other sources, such as user's manuals and tool demos, but more effort is used per each tool at this time. In addition, classification was specified a little.

Borland CaliberRM

Borland CaliberRM is an automated requirements management system intended to be used in the development of software projects. CaliberRM is designed for large and small product development teams seeking to improve the quality of their software.

CaliberRM allows collaborative, internet-based requirements management and communication among the project teams. It provides centralized requirement data to distributed team members and allows documented discussion on requirements as well as allowing the project teams to fully define, manage and communicate the changing application or system requirements. The detailed analysis on CaliberRM can be found in Appendix 1.

Rational RequisitePro

IBM Rational RequisitePro is a requirements management tool for finding, documenting, organizing, and tracking requirements. It also includes features that aid in establishing and maintaining agreement between the customer and the development team regarding the project requirements.

RequisitePro is designed for multi-user environments. It features integration of Microsoft Word and a requirements database. Software project teams can gather, enter and manage requirements within their documents or in a database. In Appendix 1, there is detailed analysis on Rational RequisitePro.

Telelogic DOORS/ERS

Telelogic DOORS/ERS (Enterprise Requirements Suite) is an integrated, Requirements Management suite designed to capture, link, trace, analyse and manage a wide range of information to ensure a project's compliance to the specified requirements and standards. DOORS is a tool primarily for large organisations which need to control complex sets of user and system requirements with full traceability.

DOORS contains functions for providing a change control mechanism and also records all changes made to objects and attributes. It can also store user-defined baselines, so that it is easier to view all modifications, additions and deletions between versions for each milestone. Appendix 1 summarises the analysis of Telelogic DOORS/ERS requirements management tool.

The following table (see Table 4) summarizes the detailed analysis of three RM tools. The number of features is plentiful in every tool, but the most extensive tool is DOORS. RequisitePro and CaliberRM are almost equal in extent between them. Moreover, DOORS and CaliberRM are meant to be used in medium and large projects and organizations, whereas RequisitePro's scope seems to be more small and medium projects.

Table 4. Summary of the detailed analysis

General criteria	CaliberRM	RequisitePro	Telelogic DOORS
Usability, Simplicity and Customization			
Multi-platform support	-	-	++
Tool integration	++	++	++
Web access	+	+	+
Access control	++	++	++
Information sharing	+	+	+
Simultaneous use	++	++	++
Specific criteria	CaliberRM	RequisitePro	Telelogic DOORS
Requirements identification	++	+	+
Req. classifying and viewing	++	++	++
Formats	+	+	++
Change management	++	+	++
Traceability	++	++	++
Document importing	+	+	++
Document generation	++	++	+
Tailoring and Extensibility	+	+	+

Classification:
++ = the tool supports this criterion
+ = the tool supports this criterion in some extent
? = evidence about support can not be found or it is unclear
- = the tool does not support this criterion

DOORS is the only tool, that supports multiple platforms, whereas CaliberRM and RequisitePro only run under MS Windows. In every tool, integration capabilities seem to be good, likewise access control. CaliberRM does nicely in requirements identification compared to the other tools, whereas DOORS leads on document importing and different format utilization. CaliberRM and RequisitePro seem to be equal in strength and they have no critical flaws, with the exception of the multi-platform support, therefore they appear to be a safe choice for requirements management. DOORS seems to be more massive; it offers more features than the others, and it comes with the highest price tag, too.

Generally speaking every tool has limited web access, so installation is required to gain the full features. Moreover, information sharing is a little bit weak in every tool as well as tailoring and extensibility. Support for tool integration, access control and simultaneous use is good in every tool, as well as requirements classifying, requirements viewing and traceability.

5. DEVELOPMENT OF THE PLUG-IN PROTOTYPE

This chapter introduces the plug-in prototype development phases as follows: First, the starting point for the development process and the role of the plug-in are presented, and reasons for choosing certain software tools for this tool integration are defined. Second, requirements for the plug-in are described, and plug-in implementation environment is presented. Third, the plug-in architecture and its implementation are described on a general level.

5.1. Starting Point for Development

When developing software products in a collaborative environment, it is quite common that requirements for the product are determined with the customer by one organization and these requirements are then distributed, for instance, by email to the subcontractors or other organizations that are connected to the project. The purpose of this plug-in prototype development was trying to provide an access to the requirement data for all team members in the project, thereby no longer having a need to send the requirements specification documents, etc. by email to the other development parties. For example, a plug-in could connect to the RM tool repository, get the original requirements from the repository and provide them in a software development environment so that the requirements are always exact and software developers can be up-to-date. This will reduce hassle with different tools when developing software.

Another objective of this development was to improve the traceability between different project artefacts, such as requirements and source codes. This kind of Simple Links Traceability has been researched for dozens of years and it is used quite broadly in the industry. The problem is not the lack of traceability but the lack of traceability between different manufacturers' tools. Integrations and traceability between the project phase artefacts (e.g. requirements, design, code files) with a single manufacturer's tools already exists, but the fact is that organizations in collaboration usually have different manufacturer tools in their development. Thus, one objective of this plug-in development is to take the first step toward the constructing the traceability between different manufacturers' tools and to study what issues need to be taken into account when integrating tools in this way.

The Eclipse Platform was chosen as an integration platform where different software tools can be integrated. The main reason for using Eclipse as the integration platform is that it is especially developed for integrating software development tools by using its clearly defined plug-in based framework. For that reason, it provides a suitable toolkit by name "Plug-in Development Environment" to implement the integration. In addition, Eclipse is an open source tool continuously developed by a large community of industrial companies and it is well known, reliable and free of charge. More information about Eclipse is given below (see section 5.4).

IBM Rational RequisitePro was chosen to tackle the requirements management issues in the case study, because some of the Merlin industrial partners are using it in their projects and wanted it for the detailed analysis. It also has a sufficient number of features and is meant to be used in small- and medium-sized projects, so it seemed to be a suitable RM tool for our use. In addition, a goal in the Merlin project is to

define and develop the tool chain infrastructure, which integrates tools from different phases of project lifecycle, and Telelogic Synergy/CM was chosen as the configuration management tool for the tool chain. Telelogic DOORS was another alternative for the RM tool, but because it is common in collaboration that the organizations have software tools from different vendors, we decided to purchase tools from different vendors and try to integrate them among themselves. Thus, our case study tried to simulate the challenges of real collaborative development as well as possible.

5.2. Role of the Plug-in

Under the circumstances, plug-in developed here integrates Rational RequisitePro with the Eclipse IDE enabling software developer to easily access the requirements data by using the Eclipse IDE only for his work. Thus, a plug-in should provide a suitable and simple view to the requirements for every SW developer in a project. The need for this kind of a plug-in is the lack of direct link from RequisitePro requirements to the source code files. This link is useful during the implementation phase, but especially in a later phase when automated test cases against the requirements will be run. Via the plug-in, selecting the correct code files related to the requirement under test by using traceability information between requirements and source code files could be easy and thus it is easier to ensure that all requirements are implemented and tested. In addition, the plug-in also ensures that all requirements of the product are met, and helps tracing what source code files must be modified if a certain requirement is changed.

In the Merlin project, Eclipse Platform will be used as a platform where different tool vendors' tools are integrated in such a way that it formulates a working solution of the whole tool chain infrastructure (see section 3.5 for more information). That means the plug-in prototype will be further developed and extended later, and this version is a trial only on how this kind of integration can be done and what issues needs to be taken into account when developing an Eclipse plug-in like this. In addition, it should be taken into account, that this first version of the prototype is meant to be used in organizations only internally, not between separate organizations. In the future, this field will be extended.

Telelogic has already developed a plug-in for Eclipse that integrates Synergy/CM to the Eclipse IDE workspace. Synergy/CM is a configuration management tool that takes care of source code items and their different versions. Source code files are stored to the central repository and Synergy/CM - plug-in provides an interface for all required functionality of Synergy/CM to handle configuration management issues.

The plug-in prototype should allow such features to the Eclipse user. However, ReqPro will not provide all RequisitePro functionality to the Eclipse user, but only those that are important especially for an SW developer. For instance, the plug-in should help and simplify the developers' work by offering particular views to the requirement data, such as a list of software requirements and their attributes, traceability matrix (traceability from SW requirements to source code files) and, for example, dependencies between different product requirements like product features and software requirements. The plug-in should maintain traceability links between requirements and source codes, because RequisitePro does not offer that kind of a

feature. The detailed requirements for the plug-in are defined in the following section.

The functionality of the plug-in is tested in a case study described in the next chapter (see Chapter 6). The case study focuses mainly on the evaluation of the RequisitePro functionality, but also integration to Eclipse via this plug-in prototype is evaluated.

5.3. Requirements for the Plug-in

When specifying the tool integration from the software developer viewpoint, the basic question is what a software developer needs to know about information presented in the RM tool. In addition, at least the basic functionality from all of these software tools that will be integrated had to be known. We familiarized ourselves with the tools by studying a couple of learning projects with every tool including RequisitePro, Eclipse and Synergy/CM.

Requirements for the tool integration between RequisitePro and Eclipse were defined and after that, the Merlin industrial partners were invited to join in the tool chain infrastructure workshop, where requirements for the tool chain infrastructure were specified and agreed on. In addition, we arranged a brainstorming session with the industrial partners, where detailed requirements for the integration between RequisitePro and Eclipse were defined.

The following list introduces requirements for the Eclipse plug-in integrating RequisitePro to the Eclipse. As mentioned before, the requirements for the plug-in are defined from the software developer's point of view.

1. **Getting Requirement Data from RequisitePro Database and Bringing Them up in the Views**

The plug-in connects to the RequisitePro and gets the requirement data from the tool database. This data is collected and presented as appropriate views in the Eclipse IDE workspace. The view can be, for instance, a table of requirements and their attributes and there can be a traceability matrix, too. The plug-in must be able to present the following information in appropriate views:

- Software Requirements and their Attributes (ID, Name, Description, Priority, Status)
- Product features related to requirements
- Stakeholder Request related to requirements and features
- Traceability information between requirements and source code files

2. **Managing Traceability Links from Requirements to Source Code Files for an Appropriate Product Version**

The traceability links between the requirements and source code files must be managed in some way. It must be possible to add, remove and modify traceability links between the requirements and files. This traceability information should be stored in some permanent database, e.g. in the repository. It must also be possible to add/remove source code files to/from the traceability database.

The traceability form used in this case should be the so-called Simple Links Traceability, as it is easy to understand and implement, it is quite commonly used in the industry, and many other RM tools support it. Moreover, since RequisitePro is meant to be used in small to medium-sized projects, this kind of traceability seems to be suitable in this case, because there is no risk that the traceability matrix grows uncontrollably large.

3. Search All Source Code Files that Relate to the Appropriate Requirement and Vice Versa

When choosing one requirement in the view, the plug-in shows a list of code files that relate to the appropriate requirement, thereby making it easy to open and edit the correct source code files. In addition, it should be possible to list all requirements that relate to one source code file.

4. Operates over the Organization's Intranet

In software projects, it is common that the source code files and requirement data are stored in organization’s repository; thereby making them available for all team members. Thus, it must be possible that the plug-in can operate over the organization's intranet providing an access to the RequisitePro database server.

5. Installation and Introduction

Installing the plug-in to all team members must be an easy and quick process. During the installation, the user should be asked at least the requirements database location, so that a connection to the database can be established easily. The plug-in should provide a short user manual, which helps the user in installation and introduction issues.

5.4. Eclipse Platform Architecture

The Eclipse Platform is an open source tool that is designed for building integrated development environments (IDEs). The Eclipse Platform’s principal role is to provide the tool providers with mechanisms to use and rules to follow that lead to seamlessly integrated tools. It also provides useful building blocks and frameworks facilitating the development of new tools. Eclipse operates under an open source paradigm, with a common public license that provides royalty free source code and worldwide redistribution rights for tool developers with flexibility and control over their software technology. [47]

Eclipse based tools give developers freedom of choice in a multi-language, multi-platform, multi-vendor environment. Eclipse provides a plug-in based framework that makes it easier to create, integrate and utilize software tools. The Eclipse Platform is written in the Java programming language and comes with extensive plug-in construction toolkits and examples. [47]

The Eclipse Platform is designed to meet the following requirements:

- Support the construction of a variety of tools for application development

- Support an unrestricted set of tool providers, including independent software vendors (ISVs)
- Support tools to manipulate arbitrary content types (HTML, Java, C, JSP, etc.)
- Facilitate seamless integration of tools within and across different content types and tool providers
- Run on a wide range of operating systems, including Windows and Linux
- Capitalize on the popularity of the Java programming language for writing tools

The Eclipse Project is an open source software project providing a platform for the development of highly integrated tools. It is composed of the following three subprojects; Platform, Java Development Tools (JDT) and Plug-in Development Environment (PDE). [49]

The architecture of Eclipse Platform is presented in Figure 6. The Eclipse Platform is built on a mechanism for discovering, integrating and running modules called *plug-ins.* A tool provider can write a tool as a separate plug-in that operates on files in the workspace and surfaces its tool-specific user interface in the workbench. When Platform is launched, the user is presented with an integrated development environment composed of a set of available plug-ins. The quality of the user experience depends much on how well the tool integrates to the Platform and how well various tools work with each other. [49]

A small kernel called the Platform Runtime handles the start-up and actually, all of the Platform's functionality is located in the plug-ins. Eclipse Platform Runtime handles start up, when the plug-ins installed are discovered, extensions and extension points are matched up, and a global plug-in registry is built. Each plug-in has its own Java class loader and they are only activated as needed. That procedure helps avoiding long start up times. [49]

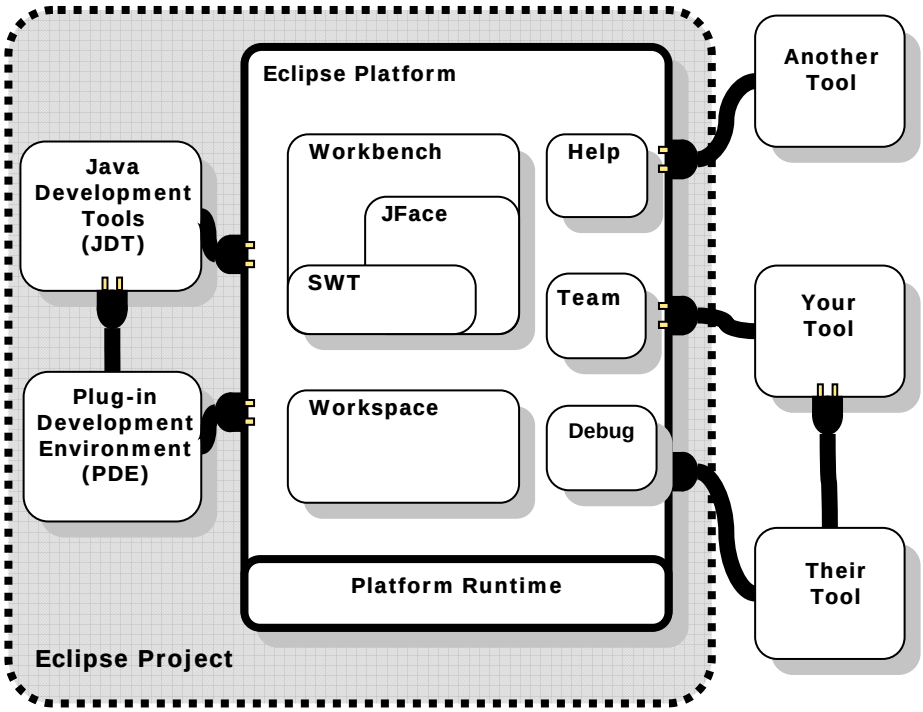


Figure 6. The Eclipse Platform.

A plug-in is the smallest unit of Eclipse Platform function that can be developed and delivered separately. As mentioned above, almost all functionality of the Eclipse Platform is located in the plug-ins. The plug-ins are written in Java and they contain Java code libraries and other files. Nowadays, every plug-in is packed as a Java library file, called a Jar-file. A plug-in is a part of the Eclipse Platform but it can be thought of as a separate Java application that can be distributed separately and can be attached to Eclipse by using a specified interface. [49]

Every plug-in has a manifest file, that declares the interconnections to other plug-ins. The interconnection model declares the extension points and extensions to the other plug-ins. An extension point is a named entity for collecting contributions. As can be seen in figure 6, the plug-ins can also be integrated among themselves. Normally, a small tool is written as a single plug-in, but a complex tool may contain several plug-ins which declares its functionality. [49]

Developing plug-ins in Eclipse is well guided. Eclipse includes a Plug-in Development Environment (PDE) that contains a wizard to start the plug-in projects with the basic functionalities and interfaces to the Platform. PDE also includes a wizard for creating installable Jar-files, thereby making every plug-in quite easy to distribute and install to the other parties' desktops.

5.5. Implementation Alternatives for the Plug-in

There are three possible approaches to implement the connection between Eclipse IDE and RequisitePro; the first one is using RequisitePro API, the second is using direct link to the requirement database through JDBC-driver and the third approach is using RequisitePro baseline files as a source of requirements information.

The first approach is using RequisitePro API as an interface to the requirements data. As RequisitePro provides an Extensibility Interface to get information from the RequisitePro, it would be natural to use it as an interface between RequisitePro and Eclipse IDE. This alternative provides a defined interface to the requirement data from outside of RequisitePro, when permissions and access rights to the database handling and other activities are given. Component Object Model (COM) implements the extensibility interface, thus any given COM enabled language, such as Visual Basic and C++, can be used for writing tool extensions. Figure 7 clarifies this architecture.

The problem in this solution is that COM-based interface needs to be handled by the above-mentioned programming languages, but Java is not directly enabled. However, a couple of third party tools exist, which can convert COM-objects to the Java classes, when COM-interface could be used by Java. In that case, the interface to the RequisitePro could become more complex. At first, the plug-in uses generated Java-classes which handle COM-objects which, finally, get information from the saved RequisitePro views or use RequisitePro’s query language to get data from the database. In addition, although RequisitePro provides documentation for the COM interface, all code examples for extensibility interface are written in Visual Basic, which also partially makes it more difficult to use this interface by Java.

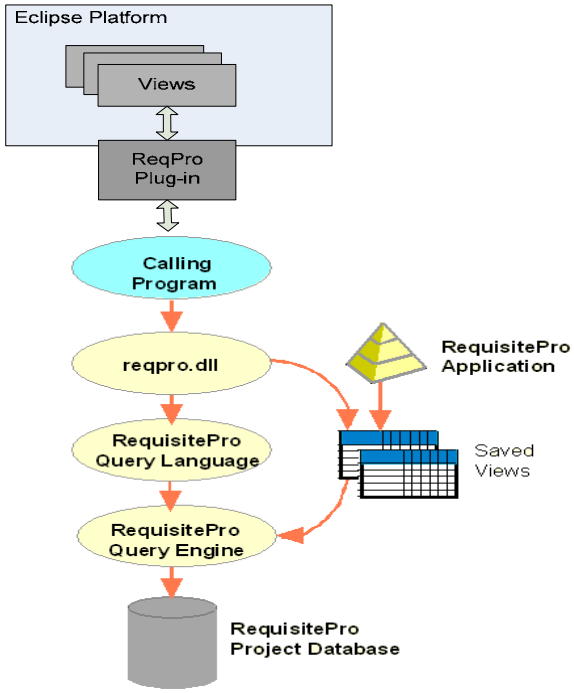


Figure 7. Conceptual architecture of the first alternative.

The second approach is using the RequisitePro database directly by a plug-in through the JDBC-driver. This solution is quite straightforward and simple by its architecture, as the following figure illustrates (see Figure 8). It is quite easy to read data from the database when its structure is known but this solution does not provide a writing access to the database. For that reason, this solution can only be used if there is no need to write data to the RequisitePro database from the Eclipse. Moreover, in this case, the plug-in can be easily modified to understand different databases, thus it should be easy to use this kind of a plug-in with a different RM tool with light modifications.

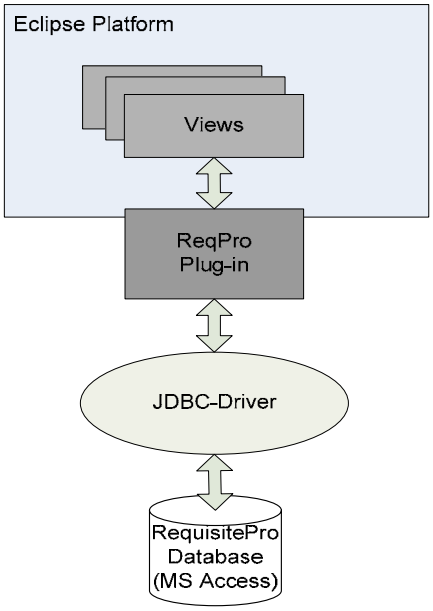


Figure 8. Conceptual architecture of the second alternative.

The third alternative is getting requirements data from the requirements baseline. Baselining in RequisitePro is implemented by using RequisitePro Baseline Manager that creates a set of XML files, where all the project data are stored. These files are stored in a user-defined directory. However, this method was refused in the beginning of the ReqPro design phase, because the data in the baselined RequisitePro project is stored in several XML files, due to which searching for pieces of information seemed to become quite difficult and complex to implement. Further, understanding the structure of the XML files is much more difficult than in the case of databases, thereby increasing the plug-in prototype development time significantly. In addition, searching information from the XML files is slower than making queries to the database. With RequisitePro baseline manager, it is possible to transform the baselined XML-shaped project to the normal project format, when all project information is generated from the XML files to one database. Thus, the project baseline is stored in an intelligible format and there is no need to handle the XML files with the baselined project.

In design meetings and workshops with the Merlin partners, we realized that there is no need in this case to change information stored in the RequisitePro requirement database, but reading and viewing the requirement data is enough. These baselined requirements can then be traced to the other product artefacts, and this traceability information (e.g. between requirement and code) can be stored in a separate editable database. Thus, the second alternative solution provides an adequate functionality for our needs.

Under the circumstances, we concluded to use the second approach to implement the plug-in in this case, where the ReqPro plug-in is a prototype that only provides the basic functionality, and it is the first trial of the plug-in development. However, if the ReqPro - plug-in is designed from project management point of view, for instance, there should be a possibility to change the requirements from outside RequisitePro and then the first approach is applicable.

5.6. Design of the “ReqPro” Plug-in

The conceptual architecture of the plug-in is introduced above in Figure 8, and this section introduces, how the *ReqPro* plug-in itself is designed. It should be noticed that ReqPro plug-in is only a prototype, so the architecture is quite generous. Moreover, when testing some new idea, such as this plug-in, it is not necessarily possible and meaningful to design the whole solution in advance, but the solution may be implemented by performing some tryouts. For that reason, attention was yet not paid to the performance and usability issues. Figure 9 illustrates the architecture of ReqPro on a general level.

The plug-in design started by outlining a link to the requirements database. The Database Handler takes care of the database connections; it reads data in the RequisitePro database and it does the read and write -operations to the Traceability Database. As the database name defines, all traceability information between requirements and source code files are stored in the Traceability Database. The Database Handler uses the JDBC-driver to connect to the database and queries are written in the SQL query language. Both databases are located in the central repository.

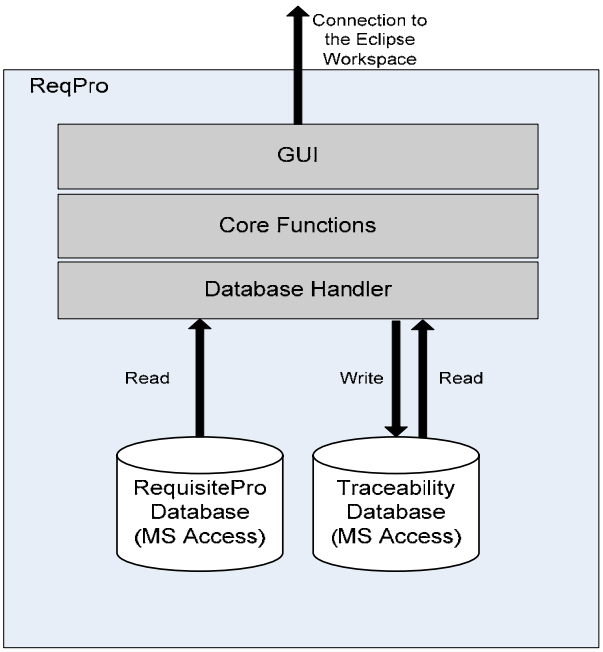


Figure 9. ReqPro Simplified Architecture.

In order to enable the plug-in to do certain operations, such as managing traceability information between requirements and source code files, it requires an engine that is here called Core Functions. This element also transmits requests from GUI to the Database Handler and transmits data from the Database Handler to GUI.

GUI (Graphic User Interface) contains classes that define an interface to the user. Its main task is to convert data to the representative form. GUI consists of different views in the Eclipse Workspace, such as the Requirements Table and the Traceability Matrix, where the user can do certain operations. For example, the Traceability Matrix view may provide functions to manage the traceability information between the requirements and source code files. The Requirements Table lists all requirements and meaningful attributes in the table form. GUI may also include different views, if they are seen important and meaningful during the prototype development. Some kind of a traceability tree view may be useful for the software developer, when the developer wants to see the relations between the software requirements and the product features or stakeholder requests.

5.7. Implementation of the ReqPro Plug-in

In this section, the implementation of the ReqPro is introduced. The purpose is not to explicate the source code of classes and components but to introduce how the design plan is realized and what the ReqPro plug-in looks like in the Eclipse IDE. We go through the ReqPro implementation by introducing a couple of screenshots and explaining what they mean and how they are meant to be used.

The installation and introduction of a ReqPro plug-in is quite a straightforward operation. The plug-in is packed as a ReqPro.zip file, which includes the plug-in itself (ReqPro.jar) and a readme.txt file, where the ReqPro installation and introduction directions are presented shortly. The ReqPro.jar includes the plug-in class-files defining the plug-in functionality, the JDBC-driver and other required files, such as plugin.xml file, icons, etc.

When the ReqPro plug-in is launched for the first time, the RequisitePro database path must be defined, thereby establishing the connection to the database, loading the requirements data, and displaying it in the views. The database path is defined simply by choosing the RequisitePro menu in the Eclipse IDE tool menu and there clicking "Define database path". After that, the user is asked to type the database path (See Figure 10) in the UNC-format (Universal Naming Convention).

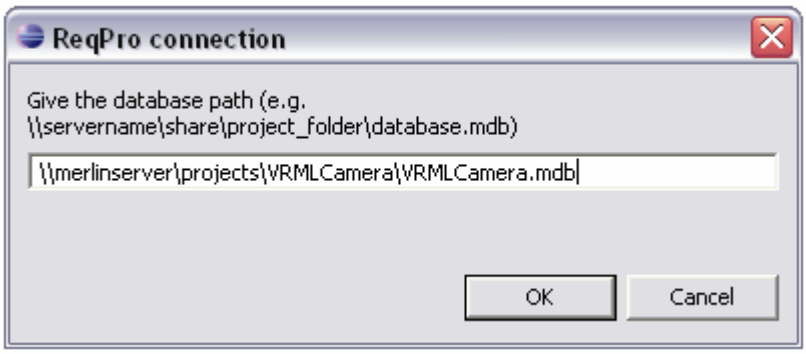


Figure 10. Defining Database Path.

When the database path is defined correctly and connection to the RequisitePro database is established, the ReqPro perspective is ready for use. Figure 11 represents the ReqPro perspective in Eclipse IDE workbench.

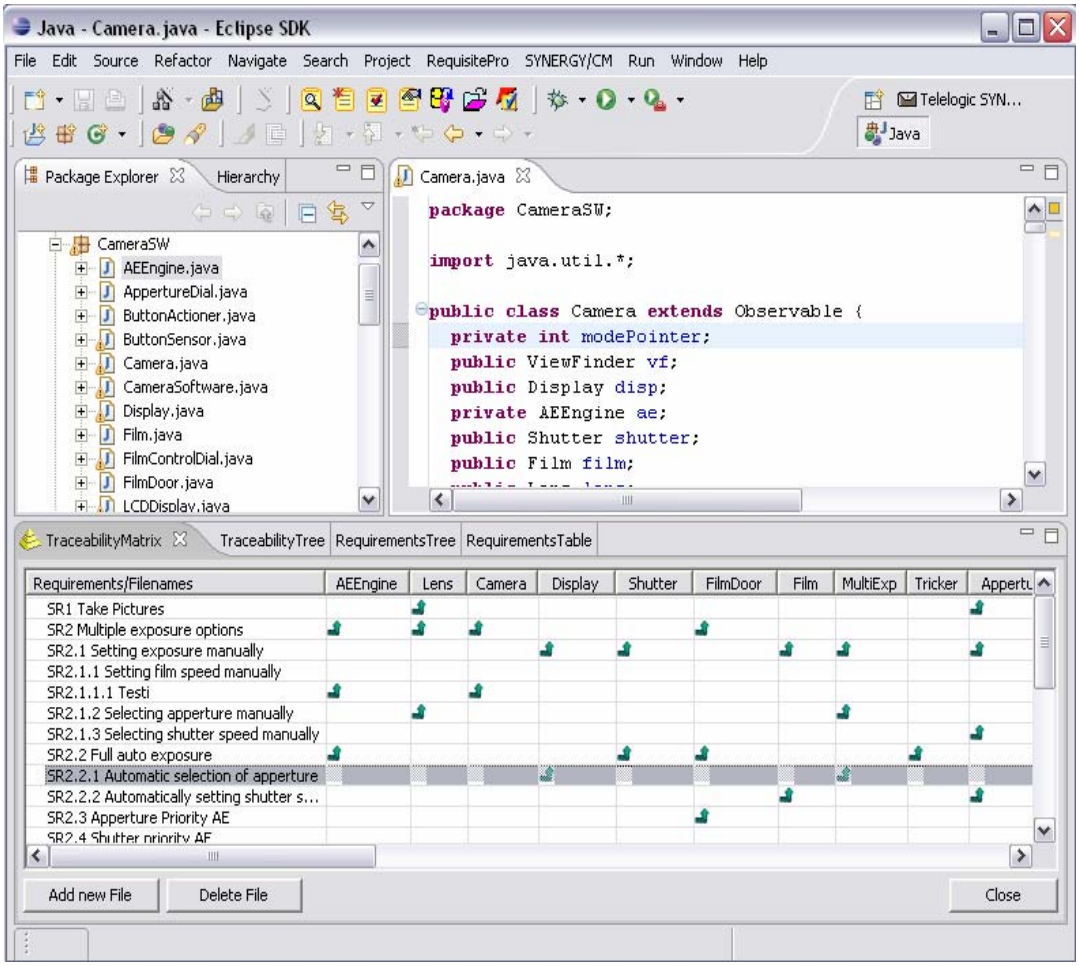


Figure 11. ReqPro Perspective in Eclipse IDE.

The lower half of the Figure 11 shows the Traceability Matrix view, where the traceability information between the requirements and source code files is managed. The traceability links can be added and removed by clicking a certain cell in the matrix. In addition, source code files can be added and removed from the view by using the appropriate button below the view. The traceability information is stored in the traceability database.

Figure 12 is a Requirements Tree view, where all the software requirements are listed in a tree formed view. In the Requirements Tree view, it is possible for example, to show the source code files, which are traced to the appropriate requirement. Moreover, it is possible to read the requirement description, show a list of the traced features and the stakeholder requests. This view is a part of the Eclipse workbench, although it is presented here separately to save space.

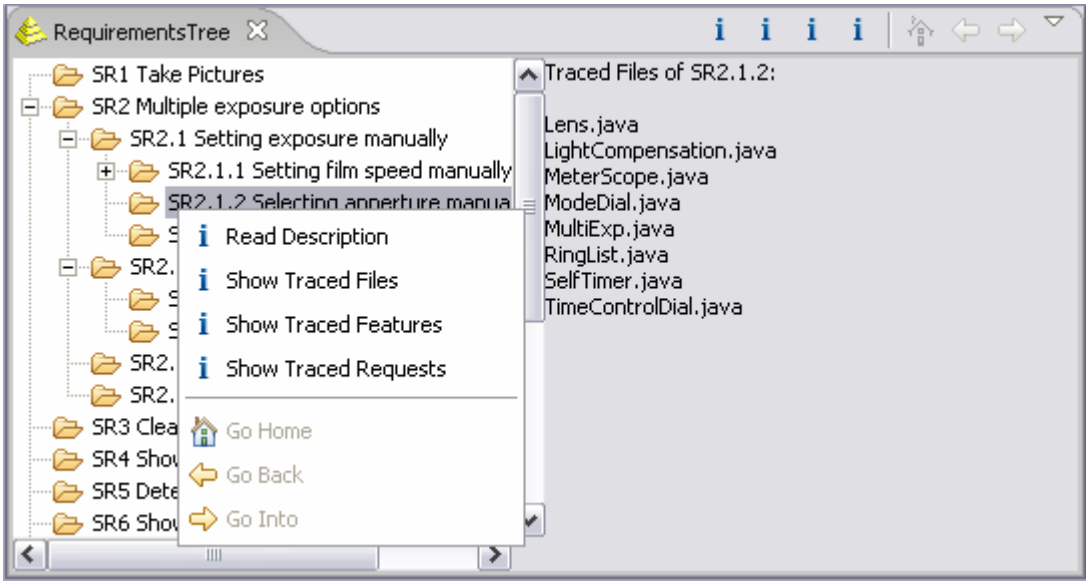
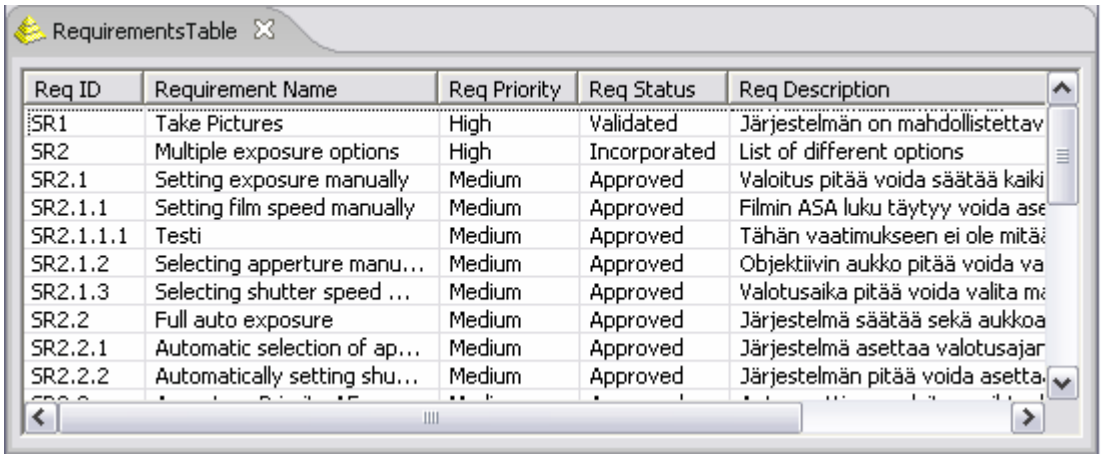


Figure 12. Requirements Tree View.

The next figure (see Figure 13) is a simple list of all requirements and a set of their attributes. This view is a part of the Eclipse workbench, too.



Req ID	Requirement Name	Req Priority	Req Status	Req Description
SR1	Take Pictures	High	Validated	Järjestelmän on mahdollistettav
SR2	Multiple exposure options	High	Incorporated	List of different options
SR2.1	Setting exposure manually	Medium	Approved	Valoitus pitää voida säätää kaiki
SR2.1.1	Setting film speed manually	Medium	Approved	Filmin ASA luku täytyy voida ase
SR2.1.1.1	Testi	Medium	Approved	Tähän vaatimukseen ei ole mitä
SR2.1.2	Selecting apperture manu...	Medium	Approved	Objektiivin aukko pitää voida va
SR2.1.3	Selecting shutter speed ...	Medium	Approved	Valotusaika pitää voida valita ma
SR2.2	Full auto exposure	Medium	Approved	Järjestelmä säätää sekä aukkoa
SR2.2.1	Automatic selection of ap...	Medium	Approved	Järjestelmä asettaa valotusajan
SR2.2.2	Automatically setting shu...	Medium	Approved	Järjestelmän pitää voida asetta

Figure 13. Requirements Table View.

The last figure (see Figure 14) of the ReqPro plug-in presents the Traceability Tree view, which clarifies the whole traceability path in a tree-formed view from the Stakeholder Requests (STRQ) to the Product Features (FEAT) and further to the Software Requirements (SR). In this view, it is also possible to read the requirement description, show all the traced files, features and requests of a certain requirement. This view type is adapted from the RequisitePro view types.

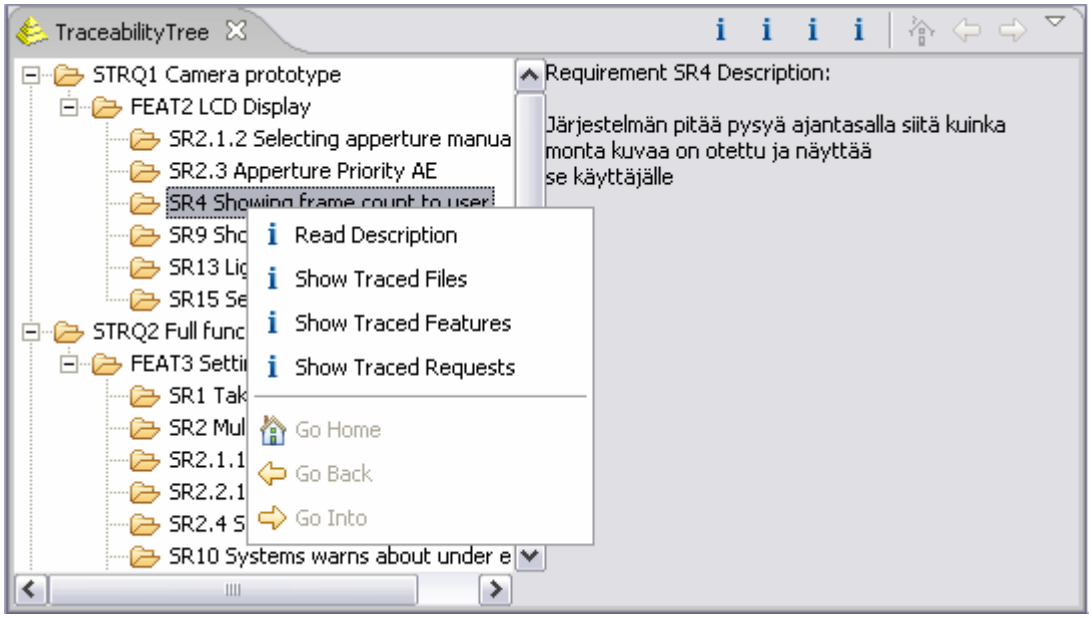


Figure 14. Traceability Tree View.

It should be noted, that the ReqPro plug-in does not manage the requirements in any way, but it only gets them from the RequisitePro database and brings them to the appropriate Eclipse view. The RequisitePro database is not edited at all; only read operation is done. Meanwhile, the traceability information is stored in a separate editable database.

6. CASE STUDY: THE VIRTUAL CAMERA PROJECT

This chapter introduces a case study which is used to evaluate RequisitePro and its integration with Eclipse IDE. In addition, the Telelogic Synergy/CM tool and its integration with Eclipse IDE is tested and evaluated too, but this subject is not described in detail in this thesis.

The structure of this chapter is as follows: first, the background for the case study and an overview of the enclosed tools are introduced. Then, arrangement and scenario of the study is presented. After that, experiences of the use of the tool, improvement proposals for tools and ReqPro are discussed. Finally, a short summary on the case study is given.

6.1. Background for the Case Study

In collaboration, it is quite common that different organizations use development tools from different vendors. We recognized two different approaches, which were possible for us for the demonstration of tool chains. The first option was to study the situation where all the collaboration partners had the same information system environment. In this case, there would not be so many challenges to be tackled to get the RM and CM environment to work together. For instance, Telelogic Doors for RM and Synergy for CM could have been possible candidates for this purpose.

Anyway, we wanted to experiment more with a real world situation when different organizations want to co-operate but are using tools from different vendors for historical reasons. It means that these two different tool environments have to be unified by purchasing the same kind of tools or to use these existing tool environments and try somehow to survive with them. We selected the more complex option to study the challenges related to the situation where two organizations with their own RM and CM tools want to collaborate, but they do not want to invest on purchasing new tools. Therefore, it is important that the tools can be integrated with each other in some way.

As mentioned above, RequisitePro and Synergy/CM were chosen for the case study, because some Merlin industrial partners use both tools in the real life and they were wanted for a detailed analysis. More explanation on choosing certain tools can be found in section 5.1.

The purposes of the case study are:

- Getting information and experiences about constructing and implementing an integrated system that should help requirements and configuration management in the collaborative software development
- Studying how RequisitePro can be utilized in a requirements management process in a collaborative environment
- Finding weaknesses and problems, but also successes and good issues, from tools in this environment
- Testing and evaluating the ReqPro plug-in prototype in order to learn more about this kind of integration

6.2. Overview of Tools and Their Roles

This section gives a short description about Rational RequisitePro, Eclipse and Telelogic Synergy/CM. More information about RequisitePro's functionality and features can be found in section 6.5.1 and Appendix 1.

IBM Rational RequisitePro is a requirements management tool for finding, documenting, organizing, and tracking requirements. It also includes features that aid in establishing and maintaining agreement between the customer and the development team regarding the project requirements. RequisitePro is designed for multi-user environments. It features integration of Microsoft Word and a requirements database. Software project teams can gather, enter and manage requirements within their documents or in a database.

Web access to the requirements database is gained by installing *Rational RequisiteWeb* server to the network server. RequisitePro needs to be installed to every client computer to gain full features for the requirements management, but limited features are offered by the RequisiteWeb web access, and it does not require installation in the client computers. This makes it possible for the customers and other stakeholders to access the requirements database through the internet by a standard web browser. RequisiteWeb requires certain server configurations, an appropriate access control and security functions to the server machine. The RequisitePro's Web interface makes the requirements accessible to all team members, including remote locations or multi-platform environments.

Telelogic Synergy/CM is a task-based configuration management solution designed to help development teams work faster and easier. Synergy/CM tries to accelerate the release management process, maximize the efficiency of limited development resources that are always in demand, and unite distributed development teams. Synergy/CM offers a distributed repository, and a proven, team-oriented workflow approach to the software development. Synergy/CM is used in this case study from Eclipse IDE through the Synergy/CM plug-in. The plug-in provides all required functions for the software developer to use the Synergy/CM client for the configuration management. [46]

The Eclipse Platform is an open source tool that is designed for building integrated development environments (IDEs). It can be used to create, for example, embedded Java programs, C++ programs, and Enterprise JavaBeans. Eclipse operates under an open source paradigm, with a common public license that provides royalty free source code and worldwide redistribution rights for tool developers with flexibility and control over their software technology. Eclipse based tools give developers the freedom of choice in a multi-language, multi-platform, multi-vendor environment. Eclipse provides a plug-in based framework that makes it easier to create, integrate and utilize software tools. The Eclipse Platform is written in the Java programming language and comes with extensive plug-in construction toolkits and examples. [47]

6.3. Arrangement of the Case Study

This case study is done in VTT Electronics using the internal data network. Since this project was performed inside VTT, it only simulates a real collaborative software development project.

VTT has developed a “Virtual Camera prototype” for demonstration purposes and the software files related to this demonstration were used during this study. The camera prototype includes the following type of information:

- Requirements specification document
- Design and code files
- Traceability matrix between requirements and Java-components

During this study, the requirement items of the Virtual Camera prototype were fed into RequisitePro whereas the design and code files were imported to the Synergy/CM server. The traceability information between requirements and source code files was transferred into a Traceability Database with the ReqPro plug-in that also provides functions to manage the traceability information in Eclipse IDE.

The Telelogic Synergy/CM Server carries out software and design configuration management issues and it offers a database server, where all the CM items are stored. Synergy/CM requires a client installation in the workstations.

The installation process was started by acquiring the server hardware. Windows Server 2000 was purchased and installed in the server. After that, the Telelogic Synergy/CM 6.4 fileserver, Rational RequisitePro and RequisiteWeb server were installed and configured in the server. RequisitePro stores requirements items by default to the MS Access database, whereas Synergy/CM uses the Informix database system. The Eclipse IDE with the ReqPro and Synergy plug-ins are installed in the client computers. Figure 15 below clarifies the technical environment of the project.

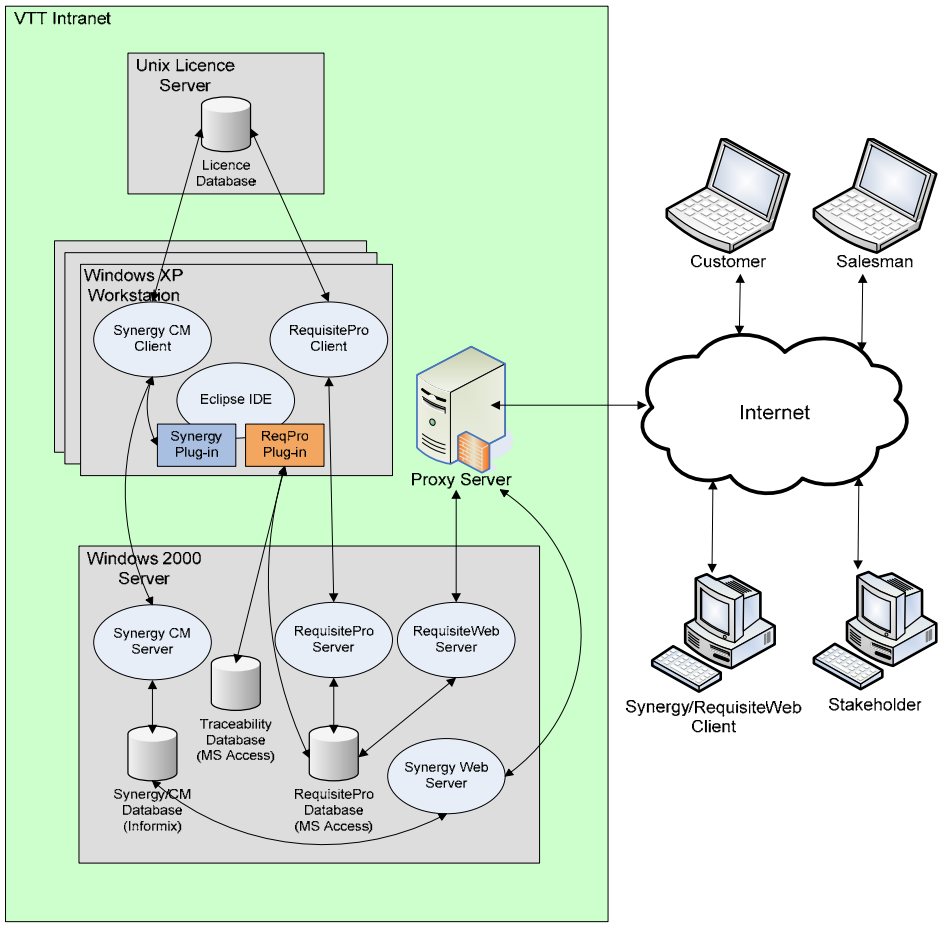


Figure 15. Technical Environment of the Project.

Both Synergy/CM and RequisitePro were purchased with a floating licence, thereby needing a licence server to store their licences. VTT has the required licence server already, where the licences were stored. The floating licences basic rule is the following: a software developer starts the Synergy/RequisitePro client software, and then the client asks the licence server for a free licence to run the software. If there are free licences in a licence pool, the client software can start and this licence is reserved to this developer. When the developer quits the client session, the particular licence is returned to the licence pool.

As mentioned above, when using Synergy/CM and RequisitePro with full functionality, the client software needs to be installed in the developers' workstations. RequisitePro only runs under Windows clients, so both tools are installed with the Eclipse IDE in the developers' Windows XP workstations, so that integration between these tools is possible. Some Synergy/CM administration operations, such as database and user management, had to be done by using a server machine.

The use cases are used to help analyzing RequisitePro tool utilization and integration with Eclipse and Synergy/CM. This analysis is done against the scenario described in the next section and the use cases are attached to a certain phase of the development process. The use cases represent such actions in requirements management which are difficult to handle in collaboration, at least without an RM tool, but they also include basic actions during the requirement management process. Examples of the use case descriptions can be found in Appendix 2.

6.4. Scenario – The Virtual Camera Project

This scenario is a fictional description of the project, where the RequisitePro and Synergy integration and their integrated use are evaluated. The purpose of this scenario is to simulate collaborative software development with the help of appropriate tools. The scenario represents a conventional collaborative embedded systems development project and its background at the moment.

A well-known digital camera manufacturer (Company 1) is planning a brand-new digital camera model. They decid to develop the first camera prototype by using virtual prototyping, thereby simplifying the planning of the functionality and physical design of the new camera model. When planning the camera by using virtual prototyping, it possible to create a fully functional virtual prototype of their camera model by using the VRML model for physical design and the Java language for the functional implementation. The product development demands embedded systems engineering skills, but Company 1 is mainly focused on hardware engineering, so it needs to acquire software engineering expertise from outside of the company.

Company 1 starts negotiations with subcontractors Company 2 and Company 3, and they come into terms as follows: Company 1 develops the hardware for the product and acts as an integrator and coordinator. In this case, Company 1 is only developing hardware for the camera and this development activity is left out of this case study. Company 2 and Company 3 divide the software-developing work in equal halves, thereby giving Company 3 the responsibility for creating the VRML model for the physical design of the camera, and Company 2 the development of the

functionality to the camera by using the Java programming language. Company 1 integrates and tests these outputs after the implementation phase.

All companies are located at physically different places, that is to say, they are geographically divided. This may create some difficulties for the development work, such as communication and other problems, like requirements management, project management and configuration management problems.

The companies have purchased certain tools to help the development work in collaboration with other companies. To support the requirements management and configuration management, Company 1 has purchased Rational RequisitePro and Company 2 has purchased Telelogic Synergy CM to support the software configuration management and it has Eclipse IDE for developing the Java applications, too. In addition, they have a ReqPro plug-in installed to the Eclipse, thereby simplifying the getting and reading of requirements in the repository during the implementation. Company 3 uses RequisitePro's web access to access the requirements database, because it only rarely requires that connection.

All companies have used the aforementioned tools for a long time, so they decide to use them in future, although cooperation between these tools can cause some problems during the development process. On the other hand, introduction of brand-new tools can create a significant additional cost for the companies. For example, these kinds of tools are quite expensive to purchase and training is expensive, too.

6.4.1. Project Phases

The project is divided into five phases and each of them has its own responsible actor. The following section gives an appropriate description of the activities and responsibilities in every phase. Figure 16 illustrates the project's collaboration activities between the companies during the development and it clarifies who is responsible for a certain project phase.

Requirements Management

Company 1 collects all the requirements and stores them into RequisitePro. They also create the traceability links needed between the requirements. After all requirements have been collected, Company 1 creates the requirements baseline and delivers the requirements baseline for companies 2 and 3. Company 1 is responsible for the requirements change management activities.

SW Design

Based on the requirements baseline, Company 2 starts the SW design and implementation and stores all the outputs in Synergy/CM. Company 2 checks the integrity between the requirements and its own design and implementation. They create the traceability between requirements and design and implementation outputs (design and code files) by establishing the RM-CM traceability matrix. When all the requirements are implemented and the traceability matrix has been updated, Company 2 baselines the outputs and delivers them to Company 1 for the integration phase.

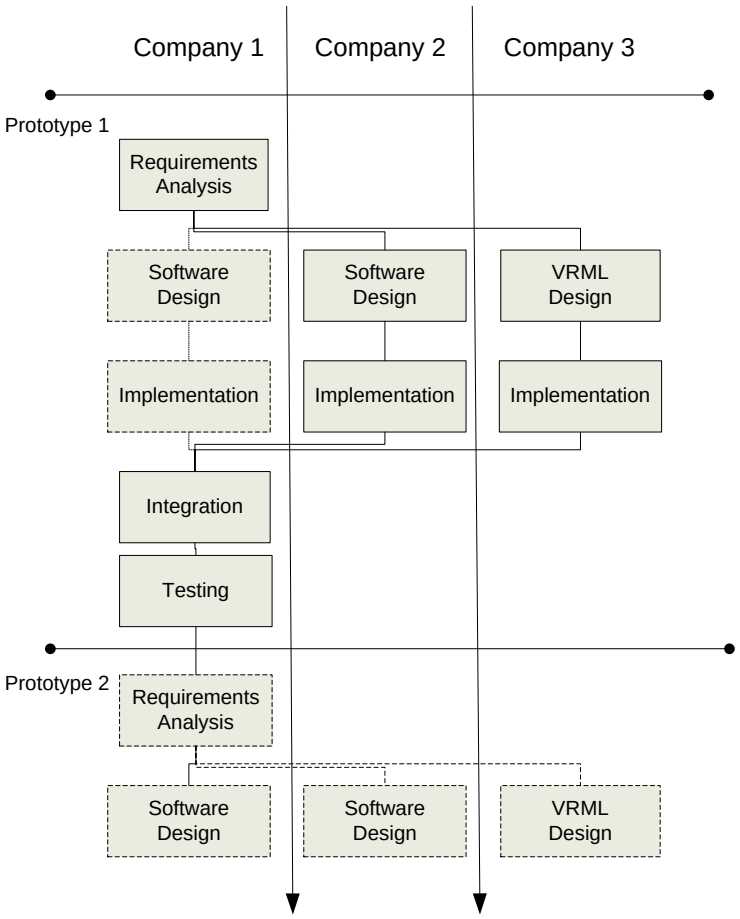


Figure 16. Collaboration Activities between the Companies.

VRML Design

Based on the requirements baseline, Company 3 starts the VRML design and implementation and stores all the outputs in Synergy CM database. Company 3 checks the integrity between the requirements and its own design and implementation. They create the traceability between the requirements and the VRML design and implementation outputs by establishing the RM-CM traceability matrix. When all the requirements are implemented, Company 3 baselines the outputs and delivers them to Company 1 for the integration phase.

Implementation

Company 2 implements the Virtual Camera functionality by using the Java language. Implementation is done in Eclipse IDE, where ReqPro and Synergy/CM plug-ins are installed. The ReqPro plug-in provides a direct link to the requirements data and it also makes it possible to trace the requirements to the source code files so that it is easier to test product requirements, because relations between these items are available.

Integration

Company 2 receives the outputs from companies 2 and 3, and stores them in Synergy/CM. Company 2 checks the integrity of the requirements and outputs from the traceability matrix and integrates the product parts needed.

Testing

Company 2 creates test cases according to the requirements, runs the test events and collects the change requests from these test events. Company 2 studies the impacts of the changes to the requirements by using the requirements traceability information. Affected requirements are modified or new requirements are created and stored in RequisitePro. Company 2 updates the requirements traceability information and creates a new requirements baseline for further development. The new baseline is sent to companies 2 and 3.

6.5. Experiences of Using Tools

RequisitePro is analysed against the criteria described in section 4.1. Besides, ReqPro plug-in is analysed against its requirements, whereas Synergy/CM is not analysed, but only its plug-in features are introduced shortly. The ReqPro requirements are described in detail in section 5.3. Evaluations and experiences on every tool are discussed in separate sections, in which text passages are headlined according to the criterion or requirement.

First, experiences on RequisitePro are discussed and evaluation is given. Then, ReqPro plug-in is analysed, and finally, Telelogic Synergy/CM plug-in features are evaluated, as they pertain to the scope of the case study. Since one focus of this case study is getting experiences about constructing the collaborative environment, also issues pertaining to this area are discussed.

A use case method is used to describe separate events during the case study, because it helps introducing operations that are done in a particular moment. The use cases described in Appendix 2 are examples of the important activities in the collaborative requirements management process. The use cases were also described for other activities, but they are not presented in this thesis.

6.5.1. Rational RequisitePro

It seems that RequisitePro is designed by keeping Rational Unified Process (RUP) in mind. Of course, it is not mandatory to use RUP, but it may offer a toolkit and methods for a more effective requirement management process. In this case, we did not use RUP, because it would have been necessary to study it well before the case study, while we did not have enough time for that. However, we used a traditional project template that creates a certain file and database layout for our requirement project.

The following figure (see Figure 17) presents the RequisitePro user interface. On the left side of the figure, there is the structure of the project presented in the tree view. The right side of the figure represents one of the many views of RequisitePro.

The view is named the Traceability Tree. The same kind of view was implemented in the ReqPro plug-in, as it was presented in section 5.7.

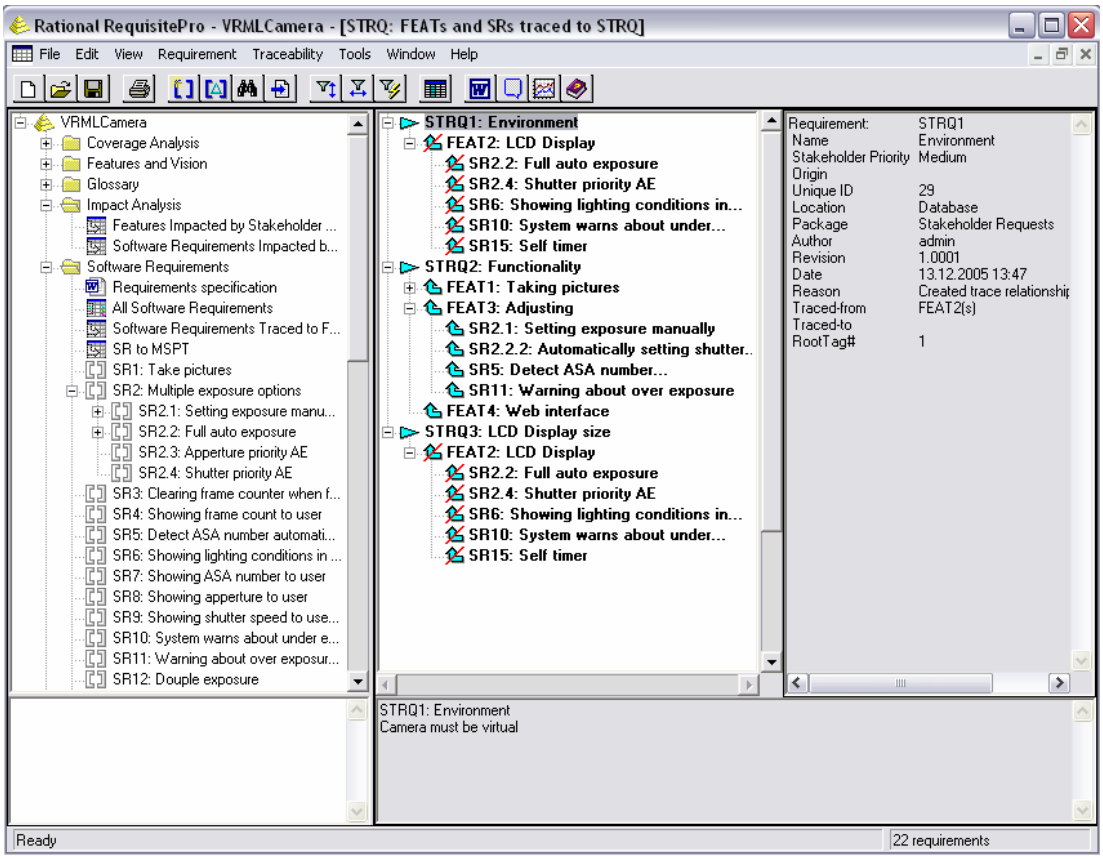


Figure 17. RequisitePro Main Window.

Introduction of RequisitePro

When Rational RequisitePro was purchased, it came first with an evaluating licence. Rational normally distributes their software via internet, but this time we wanted it by a physical medium and thus the RequisitePro software came on CD. Surprisingly it looks like a homemade package, where the software and its update file were burned on separate CDs by a normal re-writable CD drive. During the learning and tutorial projects, the connection to the licence server disappeared few times, but this problem was caused by network connection or server troubles, not by RequisitePro. Further, at the beginning, RequisitePro ran under the evaluating licence, and Rational had some problems in delivering the real licence key to us. Eventually, the permanent licence was delivered.

In addition, the RequisitePro installation in the server and clients also encountered some problems. Actually, upgrading RequisitePro from CD was not successful. We had to download a new version of the upgrade patch from the internet to finish the upgrade process of RequisitePro.

RequisitePro has no different installation software versions for the server and the workstations, but the same version of software operates in both environments. Actually, it is not mandatory to install RequisitePro to the server, but the project author and the administrator wanted to create a certain project to the server machine plainly. If RequisitePro is to be used company-wide and projects are complex, it is

recommended to use a company database. RequisitePro supports the use of IBM DB2 Universal Database, Oracle and Microsoft SQL Server. A company database offers scalability, security and administrative features for large, complex projects, but it does not have an influence on the other tool features and it requires quite a lot of configuration so that we decided to use the default MS Access database in this case study.

Creating a Project

Creating the project is straightforward; the wizard just asks the required information from the project administrator. First, the administrator chooses a project template and then the project name, directory, database and description are defined. After that, RequisitePro creates an appropriate project template and the administrator can start to work with the requirements.

Inserting Requirements to the Database

RequisitePro provides a feature that makes it possible to import the existing requirements definition documents and gather the requirements from the existing documents. However, this requires that the requirements are identified in some way in the document. We tried this feature in our case, and it seemed to work quite nicely.

We also added some requirements manually. It was done by using an imported requirement specification document, when the requirements are stored in the database, too. Another way is adding requirements directly to the database with a RequisitePro interface. It should be noted that if requirements are added directly to the database, they are not added automatically to the requirements specification document. This could lead to data fragmentation and inconsistency. This is to say, RequisitePro compels the users to use MS Word in this process. This is quite inflexible and I think it is a fault in the usability. On the other hand, this is one of the advertised features in RequisitePro and some users may like this feature.

Usability, Simplicity and Customization

RequisitePro is quite easy to use, but it has some features that may add additional tasks to the requirements management process. One example is using MS Word in the requirements management. If the user wants that requirements are also stored in the requirements specification document besides the database, all requirements need to be added to the database by using Word. I think this creates an additional task and slows down the requirement management process. On the other hand, some users may like this feature, if they have used Word in their previous requirements management work.

RequisitePro is quite easy to install and actually, no configurations are needed during the installation process.

Multi-platform Support

RequisitePro does not have support for multiple operating systems, only Microsoft Windows is supported. On the other hand, RequisiteWeb provides a web interface to the requirements and, for instance, UNIX users can use it. The problem in the use of

RequisiteWeb is its quite poor usability, it does not offer all functionality that RequisitePro has and it is meant to be used only rarely. The lack of the multi-platform support is a clear weakness in the collaborative environment.

Tool Integration

RequisitePro has some integration implemented already; for example, an interface to the Microsoft Project and to other Rational tools, such as ClearQuest and Rose, exists already. Integrations to other manufacturer tools are possible with the RequisitePro COM-based API interface. In our case, we intended to use it when integrating RequisitePro to Eclipse IDE, but the COM interface cannot be used directly by Java, so we decided to use another way to access the requirement data.

Web Access

RequisitePro has web interface, which is named RequisiteWeb. It offers almost all the same functionality as RequisitePro, but moving requirements within the classification tree is not supported. The usability is not very good by default, but the web interface can be customized by editing a CSS file that may help in the usability issues.

The web interface of RequisitePro makes it suitable for projects where the various users involved are spread all over the world. For example, customers or other stakeholders can use it, as it is not necessary to install the RequisitePro application in their workstations. Among others, it is possible to access, query, modify, create requirements and requirements documents, and manage traceability between requirements through the RequisiteWeb interface, within the limits of access rights and permissions of the user.

Access Control

The access control in RequisitePro is nicely implemented. RequisitePro provides role and project-based access control, and in addition, every user can have special access rights. Every user has a user account and a password which also works with RequisiteWeb. The project administrator can restrict the users' rights if it is seen necessary. For example, it should not be possible for a customer to edit the software requirements so their rights are restricted to read-only permissions.

Information Sharing

There is a discussion possibility in RequisitePro. It also provides email notifications for new messages, which are pointed to the appropriate person. Discussion threads can also include associations to the requirements, which improve the utilization of discussions.

Simultaneous Use

We purchased RequisitePro with floating licence which provides a possibility for many users to use RequisitePro, one user at a time. That is to say, we only have one

operative licence and thus, it was not possible to test the simultaneous use of RequisitePro.

Requirements Identification

RequisitePro supports this criterion well. It supports the requirements identification and prioritization and it provides a large scale of attributes for every requirement. In addition, the user can create these attributes. All requirements are identified by a unique ID-number and a prefix for every requirement. (For example, product feature = FEAT, software requirements = SR, etc.)

Requirements Classifying and Viewing

In RequisitePro, it is possible to classify requirements in many ways. For example, the raw customer requirements can be added to the database as product features and requirements that are more detailed are named software requirements. There are also at least the following requirement types: use cases, stakeholder requests and testing requirements. The types classify the requirements, but they can also be classified based on their attributes. Creating of different requirements views is well supported in RequisitePro, and in addition, requirements can be ordered hierarchically to the parent and child requirements.

Formats

RequisitePro does not support graphical or model based description possibilities for requirements; only textual descriptions are possible. The use case descriptions are also only in textual format. This is another weakness in RequisitePro.

Change Management

RequisitePro supports requirements change management procedure on some level. For example, if some product feature is modified, all traceability links that are associated to this modified feature are marked as “suspect”. This means these traced requirements must be checked, if they must also be modified according to the change made to the feature. Moreover, RequisitePro stores all requirements changes in the database and the history of the changes is stored, too. If some requirement is changed, the user is asked for a reason for the change and this reason is stored in the requirements history.

Traceability

Traceability between different requirements is well supported in RequisitePro. Every requirement has an owner who has created the requirement. This makes it possible to direct questions to the appropriate stakeholder, e.g. in case, where some requirement is described implicitly.

RequisitePro supports traceability to the MS Project tasks, but traceability to the design files of code files is not supported. This is a place for development of the ReqPro Eclipse plug-in.

Document Importing

In RequisitePro, it is possible to import a MS Word document to the requirement project. RequisitePro also provides a possibility of catching the requirements from an imported document, but they must be marked in some way, so that the requirements text can be identified from the document. This feature seemed to be working quite nicely.

Document Generation

Requirements specification document can be printed out, if needed. RequisitePro also provides a comparing report generation feature from the existing two baselines. Moreover, it is possible to print out a summary of a project including statistical data of the project. It is not possible to generate an extensive report of a requirement project, but only summary report generation about the project.

RequisitePro provides a possibility for baseline projects, when all project data are written into XML format. These baselines can be used for freezing requirements to a certain version of the product. The baselines can also be used as a base for later projects, when new projects are created from the baseline.

Tailoring and Extensibility

RequisitePro supports the Rational Unified Process in the requirements management process, but it can also be used in many other ways. RequisitePro can be tailored quite easily, and extensibility is implemented by providing an API, which is implemented by a COM interface.

6.5.2. ReqPro Plug-in

The ReqPro plug-in functionality is evaluated against its requirements and judgements are presented in the following sections. Every section is headlined by a ReqPro requirement name and judgements are discussed below, pertaining to the appropriate requirement. The requirements are described in detail in section 5.3.

1. Getting Requirement Data from RequisitePro Database and Bringing Them up in Views

This requirement is met at the principal rule. The requirements are got from the RequisitePro database directly by using the JDBC-driver. This is not necessarily the best way, because there is no possibility of writing information in the RequisitePro database. However, handling the database directly is adequate for us at this time, and there is no need to write anything in the RequisitePro database. It can be said that handling the database directly is a lightweight solution compared to a solution where RequisitePro API is used. If more features are needed from ReqPro, API should be used.

ReqPro provides a couple of views where the requirements data is presented, such as traceability matrix, traceability tree and requirements table. As the names

describe, the view provides different perspectives to the requirement data, when the requirements and their relationships can be presented in a meaningful way.

2. Managing Traceability Links from Requirements to Source Code Files for Appropriate Product Version

This requirement is implemented by providing the traceability matrix view, where it is possible to add and remove traceability links between the requirements and code files. In this view, it is also possible to add/remove source code files to the traceability matrix (see Figure 11).

Problems occur if the project contains a large number of requirements and code files, because then the traceability matrix enlarges incontrollably. For that reason, the traceability links should be also managed in other ways, e.g. by providing the wizard where the code file is created, and during this operation, it is possible to link the requirements and the files to each other. Another way could be the so-called drag-and-drop – method, in which the user is provided with two lists (requirements and source codes) in separate windows, and for example, the code file item can be dragged onto the traced requirement item and then a traceability link is created and stored in the database.

Traceability from the requirement to the source code would be deeper which means that a certain requirement could be traced to the exactly defined location in the source code file. For example, some source code block inside the file could be linked to a certain requirement and vice versa. In this case, it should be noted that this increases the software developer's work significantly and may not be necessarily reasonable at all.

3. Search All Source Code Files that Relate to the Appropriate Requirement and Vice Versa

The traced files for every software requirement can be searched and displayed in a list form. In this case, it would also be possible to open the appropriate files directly to the editor. Searching the requirements related to the code files is also possible.

4. Operates over the Organizations Intranet

This requirement was met, and the connection over the intranet between workstations and the database server works correctly and fine. This issue may require further study, if the database used with RequisitePro is some company database like Oracle, or if the connection to the RequisitePro is handled through its API. However, in this case, the database path definition was sufficient for establishing the connection from ReqPro to the requirements repository.

5. Installing and Introduction

Installing and introduction is an easy thing with the Eclipse plug-ins, as in case of the ReqPro plug-in. A zipped file contains a short introduction in text file and the plug-in itself that is meant to be stored into the Eclipse\plugins – folder. While Eclipse is launched, the Platform Runtime discovers the set of the available plug-ins, reads

their manifest files, and builds an in-memory plug-in registry. This procedure notifies the ReqPro plug-in and its views and actions come available finely.

Before it is possible to use the connection to the Requirements database, the database path needs to be defined. It can be done by using the RequisitePro tool menu in the Eclipse workbench (for example, see Figure 10).

6.5.3. Telelogic Synergy CM

Synergy/CM features are introduced here quite shortly, because it is not included within the scope of this thesis. However, the Synergy/CM plug-in is a part of this case study so that a short introduction about Synergy/CM and its plug-in are given.

Introduction

Telelogic Synergy was delivered in a box consisting of paper manuals and several CDs for software installation. Manuals were also included in an electronic format in one disc. The box contains discs for both the UNIX and Windows installations. The Synergy/CM Windows server comes with the Informix database system.

The Synergy CM Main Server installation encountered a couple of problems. It was not easy to find the appropriate information for the installation requirements from the installation guide. After a few unsuccessful installation attempts, we finally got on to a Telelogic consultant and got adequate information for finishing the Synergy CM Main Server installation successfully. The Synergy CM client software was very easy to install in the client computers.

The Synergy/CM plug-in for Eclipse IDE is quite easy to introduce, and it provides the most important features for configuration management operation for software engineers. The plug-in is downloaded from the Telelogic web page. The Synergy/CM client is required in order that the plug-in can be used at all.

When operating with the Synergy/CM tool, it should be noted that it is a task-based tool. This means that anything the user is going to do, there must be a described appropriate task for this operation. Synergy/CM differs in that way from the traditional file-based CM tools.

Establishing Connection

Connection to the Synergy/CM server is established with the plug-in in the same way as using the Synergy/CM client. In fact, the plug-in opens exactly the same dialog as the Synergy/CM client and the user is asked to fill in the following data fields: Username, Password, Synergy install path, Database path, and Engine host. If all given information is valid, the connection to the CM server is established.

Creating a Task

The user is compelled to create at least one open task, in order to work with Synergy/CM. Creating the task is, again, a similar operation to the Synergy/CM client and to the plug-in. The user is asked to describe certain information pertaining to the task, such as the estimated effort and a short task description. When the task is identified, the user is granted a permit to open operations with Synergy/CM.

Opening the Project

The software project can be opened in the Synergy/CM database for the reading purpose. When the project is opened, it is stored into the Eclipse workspace. It can be edited locally and store locally, but the check in – operation is not possible in the CM database. This means that opening some software project from the Synergy/CM database actually only offers read permission to the project data.

Checkout Project

This operation is intended to be used with the modifying purpose. The software project is downloaded and stored in the Eclipse workspace similarly as in the previous stage, but this operation makes it possible to check in the project to the repository. Thus, this is the normal procedure during the software development process.

Check in Project

As the project can be checked out from the repository, it must be possible to check it back into the repository. This operation stores the modified version of the project in the Synergy/CM repository, in which case Synergy automatically takes care of the version management issues.

Completing the Task

When the described operation is done according to the open task description, the task itself needs to be completed. This makes it sure, that the configuration management is done in the correct way.

6.6. Summary of Case Study

Various people could handle the requirements during the software development process in collaboration. RequisitePro helps organize the work among the requirements by providing access controls, requirement definition procedures, and communication gateways, such as email notification and discussion forum in order to facilitate communications on the requirements. In addition, RequisitePro integrates into MS Project, which helps managing project tasks pertaining to a certain requirement. RequisitePro also integrates into other Rational tools (e.g. ClearQuest, Rose and ClearCase) and API for other integrations and tool extensions is provided, too.

The web interface of RequisitePro, RequisiteWeb, makes the product suitable for projects where the various users involved are spread all over the world. RequisiteWeb offers almost all the same functionality than the “fat” client software, but moving requirements within the classification tree is not supported. This reduces the need to install RequisitePro on each user’s desktop. Although, it should be noted that RequisiteWeb, as RequisitePro itself, requires a valid licence in order to be used on any level.

The requirements classifying is implemented well in RequisitePro. When priorities, level of difficulties and other attributes can be defined for all requirements exactly, it is a clear advantage in collaborative development, because it helps managing the whole project resources.

One weakness of RequisitePro is its inability to add requirements in the Word document after their addition through the tool interface. Updating the document can only be done manually by cutting and pasting the addition. This makes RequisitePro unsuitable for projects where numerous requirements can be added directly through the tool interface. On the other hand, RequisitePro maintains a relationship between the repository and the requirements document when the existing requirements are updated in any way.

Another weakness of RequisitePro is the lack of support for multiple operating systems, because it can only be run under MS Windows. On the other hand, RequisiteWeb substitutes this lack on some level.

Further, managing the requirements changes is not implemented in a powerful way in RequisitePro. For example, if a requirement changes for some reason, all traceability links to other requirements pertaining to this changed requirement become “suspect”, and then the user needs to check all the suspect requirements manually, whether they need also to be changed. The reasons and the history of changes are recorded, but there is no other support for the requirement change management.

The main idea of ReqPro plug-in was providing a simplified view to the requirements specifications without a hassle with the requirements specification documents and different tools. My opinion is that ReqPro does this nicely, because there is no longer a need to read the requirements specifications using, for example, Word, but only Eclipse IDE and a ReqPro plug-in are needed. Moreover, ReqPro provides a possibility of reading the requirements and their associations to the other requirements, such as features and stakeholder requests, thereby offering more than traditional requirements specification documents. Thus, using ReqPro plug-in as a part of the software development process provides a couple of advantages compared to the situation where there is no such plug-in available.

Another important feature in ReqPro is the tracing possibility between the requirements and the source code files. This feature will be further developed during the Merlin project, thereby providing, for instance, views to the software tester, when the test cases can be easily designed and run against the requirements.

Using a tool supported development process in a collaborative environment demands quite a lot of the tools used. In a collaborative environment, it is important to pay attention to the access rights, as RequisitePro does. In that case, the plug-in should restrict the permissions to the requirements data thereby allowing the subcontractors read-only access to the requirement data, which is appropriate to them. The presented prototype version of ReqPro plug-in does not provide any user access control, but it is taken care of by defining the user accounts to the server machine and giving appropriate access rights to the databases for software developers. In future, this issue must be taken into account.

Overall, the tool support in the collaborative development environment is important, because it makes many issues possible which are not possible or which are difficult without appropriate tools. Moreover, using a plug-in like ReqPro simplifies and streamlines the development work, because it reduces the hassle with the set of different tools.

7. DISCUSSION

There are numerous tools on the market which are developed in order to improve and accelerate the collaborative development process. In this thesis, we have presented and evaluated a set of requirements management tools, which can be used in the collaborative requirements management. In addition, one of these tools, IBM Rational RequisitePro, was chosen for a case study, so that its functionality and utilization for requirements management could be evaluated.

In order that RequisitePro can be used effectively as a part of a tool chain infrastructure (for more information, see section 3.5), an Eclipse plug-in, by name of ReqPro, was developed. It integrates RequisitePro to the other development environment from the software engineering point of view. The ReqPro plug-in is, regardless, only a prototype of this kind of integration and is, therefore, far from ready. Under the circumstances, the whole tool chain infrastructure will be further developed during the Merlin project, because the industrial partners were quite interested in this kind of integration.

7.1. Criteria Definition for RM Tool

Developing the criteria for collaborative RM tools started with the literature study. The requirements were specified based on interviews of the industrial partners and comments from workshops with the Merlin partners.

This approach was reasonable and we got good comments and additional requirements from the inquiry and workshops for the collaborative tools. As the number of tools was huge, it was necessary to cut down the criteria a little, thereby making it possible to evaluate all tools within the limits of time and resources.

7.2. Tool Evaluation Work

The tool survey (see chapter 4) was performed based on the material found on tool vendors' web pages and in other sources in the internet. Thus, the evaluation results are not very accurate, but they are a normative base. On the other hand, within the limits of the resources, it was not possible to take all tools in evaluation use, because it would have demanded too much time and resources, which would not have been suitable and practical in this case.

Since the tools were evaluated in this way, it must be kept in mind that the material from the tool vendors' web pages is always embroidered by advertisement slogans. For that reason, we tried to find tool manuals or other such documents, where the features are defined precisely. On the other hand, we found some evaluation reports in the internet in support of the evaluation, which partly helped the work. However, it should be noted, that the evaluating work is always like giving opinions of the evaluated object by the evaluator, and thus the results are never absolute truths.

7.3. ReqPro Plug-in Implementation

In order that RequisitePro can be used effectively as a part of a tool chain infrastructure, an Eclipse plug-in named ReqPro was developed, which provided views from Eclipse IDE to the requirements data. ReqPro also provides a possibility to link the requirements to certain source code items, when it is easier to assure that all requirements are met, and thus, it is easier to, for example, run an automated test for the product developed. This kind of traceability can also help the software developer in situations, where the software requirements change for some reasons. In that case, the developer can trace it which source codes must be checked for modifications according to the requirements change.

The ReqPro plug-in prototype meets approximately all requirements set up for it, but it is not ready for industrial cases yet, so it must be further developed. Let us spend a while discussing the lacks and faults which came up in ReqPro during the development and the case study. In addition, improvement proposals for the plug-in are also presented.

At first, the connection to the RequisitePro requirements data was implemented here by establishing a direct link from the ReqPro plug-in to the requirements database through the JDBC-driver. This could be implemented more flexibly by using the RequisitePro extensibility interface. This method of implementation could provide a possibility to, for example, add, remove or modify the requirements stored in the RequisitePro database. However, we rejected this alternative, because there was no need in this case to write anything in the requirements database by the software developer. In future, there will appear a need for this kind of activity, so we would probably implement the connection to the requirements data by using RequisitePro API.

Secondly, the version of the ReqPro plug-in presented here did not give a possibility for the user to define what kind of requirements are linked to the source code files; the only alternative is the Software Requirements. There should be a possibility also to link other requirements, such as use case requirements, to the source code files. Perhaps several alternatives should be possible at the same time. In addition, tracing these items among themselves, as well as showing the traceability information, could be implemented in alternative ways.

Thirdly, ReqPro did not commit itself on the matter of requirements and source code versions. The versioning is, however, so complicated that it was not possible within these time limits to implement this feature into this version of the plug-in. Moreover, the requirements usually live and change during the project, but occasionally the requirements are frozen and a requirements baseline is created. This baseline is generally used as a base of the next iteration of the product, thus support for the continuous versioning is not necessarily required for the plug-in. One idea was that during the Eclipse IDE start-up, the user is asked what version of the product he is going to work with, and the ReqPro plug-in automatically searches the correct requirement baseline and source code versions for the user.

Finally, the links from requirements to source code items were implemented in this ReqPro version by using the direct link from requirement to source code file and storing this information into traceability database table. This was, however, quite a lightweight solution for traceability and this issue require further development, in order that the plug-in becomes more useful and safe.

7.4. Case Study

For the start, it should be taken into account that the case study in this study was performed internally in VTT, thus it only simulates a real collaborative software project. Because of that, we determined a fictional scenario, and the tools were analysed against the criteria keeping this scenario in mind. Certain persons acted in different roles during the case study and they used tools in the appropriate way, thereby making the case study as realistic as possible.

The purpose of the case study was to try to highlight some challenges and advantages of this kind of a collaborative situation. The case study was also a test jig for RequisitePro and the ReqPro plug-in. Moreover, one objective of the case study was getting experiences from these particular tools used during the study, so the results received here cannot be generalized to the all collaborative environments as such.

In order for the case study to be adequate for giving useful advises or a checklist for industrial use, we should have more experiences of the utilization of the tools. Especially, the benefits of utilization of RequisitePro, compared to the situation where an RM tool is not used, were quite difficult to analyze based on this case study, because the project was too small. Therefore, we are planning to arrange a new case study with a student group, where these tools will be further evaluated and more experience will be gathered.

The case study presented in this thesis, however, gives useful information on the plug-in development and constructing and acting in a collaborative environment, so that it can be used as a base for further development of the ReqPro plug-in and the whole tool chain infrastructure.

7.5. Future Work

Based on the above discussion, a couple of subjects for future work can be proposed as a summary. There are clearly the following subjects, which require further attention:

- further development of the tool chain infrastructure
- the ReqPro plug-in improvement
- arrangement of a new expanded case study

In order for the introduced tool chain infrastructure to cover the whole software development lifecycle, there is a clear need at least for the following tool extension: project management-, design- and testing tool plug-ins for Eclipse. While developing the above mentioned plug-ins, required interfaces between the other plug-ins must be defined clearly, in order for the traceability to be constructed and improved covering the whole tool chain infrastructure in a reasonable and effective way. In addition, the plug-ins should be designed in such a way, that it is easy to replace the integrated tool without laborious modifications for the plug-in. This improves the versatility and is a very important factor in collaboration.

The ReqPro plug-in prototype should also be further developed and improved to fit better in the collaborative environment. The plug-in prototype introduced in this thesis was designed to be used internally in organizations. Improving the usability and efficiency of the ReqPro plug-in, it should be working over the internet. This

requires tight security and access controls from the plug-in, in order for the different organizations to safely use the plug-in and to have appropriate access rights to the existing requirement data. It is also important to pay attention to the ReqPro user interface. In addition, modularity and connectivity are important issues to be considered during the further development of ReqPro.

As mentioned above, the case study performed during this study was a little lightweight, so that the arrangement of a new expanded case study is being started with a university student group. This new case study should give more information about using the introduced collaborative environment in a software development process. In addition, it may give additional improvement proposals for the ReqPro plug-in, which are not necessarily perceived yet. When the collaborative environment is evaluated with student groups and the required improvements are implemented, it would be very interesting to adopt it in a fully industrial case study with some Merlin industrial partner to gather experiences also there.

8. SUMMARY

Competition and tight time-to-market and quality requirements of products are nowadays driving companies to develop products in collaboration. Collaborative development also creates challenges; communication between parties and product- and workflow management during the development becomes more difficult. Especially the requirements management is seen as one of the most critical activities during the development process.

Using tools that support the software development can be in quite a big role in the intra-organizational development and their role even grows in the inter-organizational collaborative development. The requirements management is seen as one of the most critical processes in collaborative development, and this thesis focused on the RM tool support in the collaborative software development.

The aim of this thesis was to:

- define the criteria for collaborative RM tools
- study what RM tools are available in the market and how they meet the criteria
- take a couple of promising RM tools in detailed analysis to help to choose one of them in a case study
- construct a collaborative development environment where the tool chain infrastructure prototype can be tested in a case study
- during the construction, it came out that it was not possible to integrate the RM tool to the other tools without particular tool extension development, thus it was implemented before the case study

8.1. Workflow of the Thesis

At the beginning of this study, we made an explanation about what kind of requirements management tools are available in markets. This explanation includes defining the criteria for the collaborative RM tools and a lightweight evaluation of the RM tools against the criteria in relation to their general features and collaborative supporting features. The analysis was done based on the vendor web pages, articles and other available material found in the internet.

After this explanation, a more detailed analysis was done for three RM tools against the same criteria, but more effort was used at this time. The selection criteria for the tools chosen for the detailed analysis were the nature and superiority of the tools, the Merlin industrial partners' expectations and VTT's interests. These three RM tools were from three well-known manufacturers, and their names were Borland CaliberRM, IBM Rational RequisitePro and Telelogic DOORS.

Based on the detailed analysis and other reasons, IBM Rational RequisitePro was chosen and purchased for a case study. In the Merlin project, we are developing a tool chain infrastructure for demonstration purposes which will make it possible to integrate different development phase tools from different vendors together. RequisitePro will be one part of the tool chain infrastructure, which will cover the whole development process in the future.

In order for the tool chain infrastructure to be constructed, we purchased the Eclipse Framework which was used as a framework for the integration. The Eclipse

Framework is designed to be used as an integration framework for different software development tools, so that it was very suitable for our use.

With the help of the Eclipse Framework's plug-in development toolkit, the ReqPro plug-in prototype was developed. This Eclipse plug-in integrates the RequisitePro RM tool to the Eclipse IDE, thereby offering different views to the requirements data and removing needs to read the requirements specification document with different word processing tools. Over and above of different requirements views, ReqPro provides a possibility for the software developer to link and trace the requirements and the source code items with each other. The ReqPro plug-in will be the first part of the tool chain infrastructure, but this level of integration is only the first step toward the working solution of the tool chain infrastructure.

Finally, a case study was arranged, where RequisitePro, the ReqPro plug-in, Eclipse IDE and Telelogic Synergy were tested in a collaborative environment against a scenario, and experiences of the utilization of the tools were collected to help further development work.

8.2. Conclusions

Processes, practices and agreements are drivers for inter-company collaboration and the role of tools is to provide discipline for practices and processes as well as contribute activities for more effective collaboration. However, tools make it possible to use processes and practises in a more effective way, and in some situations, tools are, let us say, even invaluable. On the other hand, the utilization of tools may be complex, before all team members get familiar with the tools. Tools may even hinder or slow down the requirements management process at the beginning. It should be also noted that if an RM tool is used in a software project, its effective use demands that every RM team member uses it and uses it in the right way.

It seems that requirements for tool functionality are in many respects the same for the intra- and inter-company collaboration. There are just stronger needs for security, access control and controlled data sharing in the inter-organizational collaboration.

The tool survey reveals that tools contain many features that support collaboration and thereby help the development work. The problem is not the lack of tool features, but the lack of tool utilization by organizations in general. The RM tools' utilization rate seemed to be quite low according to our inquiries among the Merlin industrial partners. Reasons for this may be ignorance about the possibilities of the RM tools. Another reason may be the difficulty in choosing the correct tool or the price of the tools.

On the grounds of the case study, it can be said that the tool chain infrastructure can be constructed; even if the existing tools are not from the same manufacturer. Of course, this requires getting familiar with the tools and their interfaces, but if the same kind of framework, such as Eclipse Framework exists in the organization, it is, after all, quite easy to construct the environment where the collaborative software project can be followed through successfully and in a more streamlined way.

Finally, about RequisitePro, it can be said that its number of features is on the average level, as well as its usability. The most important advantage of RequisitePro, thinking of the collaborative development, is its web access, i.e. RequisiteWeb. By using it, there was no need to install the "fat" client software to all users' desktop, but

just the web browser, the user account for project and the connection to the requirement server are needed. As one of the weaknesses its integration with Microsoft Word could be named. For example, if requirements should be stored in the requirements specification document and the requirements database, the requirements must always be added using the Word interface. If requirements are added by using the tool interface, the requirements should be added manually on the requirements specification by using the classic copy-paste method. On the other hand, if requirements are added by using Word, RequisitePro, fortunately, keeps both the database and the documentation requirements up-to-date and synchronized with each other.

9. REFERENCES

[1] Kruchten P. (1999) The Rational Unified Process, An Introduction. Addison-Wesley, 255 p.

[2] The Merlin-project web site. (2005-07-12) URL: http://virtual.vtt.fi/merlin/

[3] Parviainen P., Vierimaa M., Tihinen M., Kääriäinen J., Takalo J. (2004) Industrial Inventory Summary (Merlin-project deliverable D1.1.3.), 37 p.

[4] Guidelines for Requirements Management (2000). United States Department of Energy, Software Quality Assurance Subcommittee, 31 p.

[5] Lormans M., vanDijk H., vanDeursen A., Nöcker E., deZeeuw A. (2004) Managing Evolving Requirements in an Outsourcing Context: An Industrial Experience Report. In: Principles of Software Evolution, 7th International Workshop on (IWPSE'04), pp. 149-158

[6] Kotonya G., & Sommerville I. (1998) Requirements Engineering: Process and Techniques. John Wiley & Sons, 282 p.

[7] Parviainen P., Hulkko H., Kääriäinen J., Takalo J., Tihinen M. (2003) Requirements engineering, Inventory of technologies. VTT Publications 508, 106 p.

[8] Sommerville I. & Sawyer P. (1997) Requirements Engineering: A Good Practise Guide. John Wiley & Sons, 390 p.

[9] Leffingwell D. & Widrig D. (1999) Managing Software Requirements – A Unified Approach. Addison-Wesley, 582 p.

[10] Rinne J. (2001) Kohti hajautettua ohjelmistokehitystä. Pro gradu – tutkielma. Tampereen yliopisto, Tietojenkäsittelytieteiden laitos, Tampere, 54 p.

[11] Paasivaara M. & Lassenius C. (2004) Collaboration Practices in Global Inter-organizational Software Development Projects. In: Software Process Improvement and Practice, 2003; 8. pp. 183-199.

[12] Öhrwall Rönnbäck A. (2002) Interorganizational IT Support for Collaborative Product Development. Dissertation from the International Graduate School of Management and Industrial Engineering, IMIE. No. 59, Doctoral Dissertation, 83 p.

[13] Igatech Systems. Requirements Design and Traceability. (2005-05-03) URL: http://www.igatech.com/rdt/rdtwalkthrough.html

[14] Tvete B. (1999) Introducing Efficient Requirements Management. In: Proceedings of the 10th International Workshop on Database & Expert Systems Applications. IEEE Computer Society, pp. 370-375

[15] CMMI Product Team (2001) Capability Maturity Model Integration. Improving processes for better products. Version 1.1. Carnegie Mellon University, 649 p.

[16] Bokhorst L., Software Development Process for Real-Time Embedded Software Systems (DESS), Task 1.8 Requirements Management method: Analysis Report (D1.8.1), 14 p. (2005-05-03) URL: http://www.dess-itea.org/deliverables/ITEA-DESS-D181-V01P.pdf

[17] Wiegers K.E. (2003) Software Requirements, 2nd Edition, Microsoft Press, 544 p.

[18] Gurd A. (2004) The Essential Guide to Effective Requirements Management within the CMMI, Telelogic White paper, 14 p. (2005-05-10) URL: http://www.telelogic.com/download/paper/CMMI_Guide_forRM.pdf

[19] Teppola S., Takalo J., Kääriäinen J. (2005) Tool chain. (Merlin-project deliverable D1.3.3.), 18 p.

[20] Hoffmann M., Kühn N., Weber M., Bittner M. (2004) Requirements for Requirements Management Tools. In: 12th IEEE International Requirements Engineering Conference (RE´04) pp. 301-308

[21] Lang M. & Duggan J. (2001) A Tool to Support Collaborative Software Requirements Management. In: Requirements Engineering. Vol. 6. No. 3. pp. 161-172.

[22] Deliver Projects Right – The First Time. Borland CaliberRM Technical Overview, (2005-05-10) URL: http://www.borland.com/caliber/pdf/crm_techview.pdf

[23] Borland CaliberRM 2005 feature matrix (2005-05-10) URL: http://www.borland.com/caliber/pdf/crm_featurematrix.pdf

[24] INCOSE RM Tool Survey, Borland CaliberRM response, (2005-05-10) URL: http://www.paper-review.com/tools/rms/response.php?vendor=CaliberRM#1

[25] Florin A., Which project management solution is right for you? CaliberRM and Active! Focus Comparison article. Falafel Software Inc, 19 p.

[26] INCOSE RM Tool Survey, Rational RequisitePro response, (2005-05-10) URL: http://www.paper-review.com/tools/rms/response.php?vendor=IBM%20Rational%20RequisitePro#1

[27] IBM Rational RequisitePro web sites, (2005-05-11) URL: http://www-306.ibm.com/software/awdtools/reqpro/

[28] IBM Rational RequisitePro Highlights, (2005-05-11) URL: http://www3.software.ibm.com/ibmdl/pub/software/rational/web/datasheets/version6/reqpro.pdf

[29] INCOSE RM Tool Survey, Telelogic DOORS response, (2005-05-11) URL: http://www.paper-review.com/tools/rms/response.php?vendor=DOORS%20ERS#1

[30] Telelogic DOORS integrations, Telelogic web pages, (2005-05-11) URL: http://www.telelogic.com/products/doorsers/integrations/index.cfm

[31] Telelogic DOORS/ERS, Dynamic Requirements Management, (2005-05-11) URL: http://www2.telelogic.com/download/brochures/ers.pdf

[32] Carrigan W., Wallenfelt B. Implementing RequisitePro into your process development methodology (2005-07-19) URL: http://www-128.ibm.com/developerworks//rational/library/05/carrigan-wallenfelt/

[33] IBM Rational RequisitePro: Detailed Description. Technical Library. (2005-07-19) URL: http://www.pts.com/wp2226.cfm

[34] IBM Rational RequisitePro Data Sheet, (2005-07-26) URL: http://www3.software.ibm.com/ibmdl/pub/software/rational/web/datasheets/version6/reqpro.pdf

[35] Welborn R. & Kasten V. (2003) The Jericho Principle, How Companies Use Strategic Collaboration to Find New Sources of Value, John Wiley & Sons, Inc, 288 p.

[36] Carmel E. (1999) Global Software Teams: Collaborating Across Borders and Time Zones, Prentice-Hall, Upper Saddle River, N.J, USA, 269 p.

[37] Hyysalo J. (2005) Summary of literature findings and industrial experiences. (Merlin-project deliverable D 1.1.4), 27 p.

[38] MOOSE-project home pages (2005-07-26) URL: http://www.mooseproject.org/

[39] Jäälinoja J. (2004) Requirements implementation in embedded software development. University of Oulu, Department of Information Processing Science, Master's Thesis, 85 p.

[40] Telelogic DOORS Microsoft Project integration, Coordinating requirements and project management, 2 p. (2005-07-26) URL: http://www.telelogic.com

[41] Rational TestManager Integrations, 26 p. (2005-07-26) URL: http://www-128.ibm.com/developerworks/rational/library/content/03July/2500/2644/ps-2644.pdf

[42] Integrating with Visual Studio .NET, "CaliberRM", Borland Software Corporation, 12 p. (2005-07-26) URL: http://info.borland.com/techpubs/caliber_rm/downloads_en/IntegratingCaliberRMwithVisualStudioNET.pdf

[43] Bokhorst L. Software Development Process for Real-Time Embedded Software Systems (DESS), Task 1.8 Requirements Management method definition (D1.8.2) , 17 p. (2005-07-26) URL: http://www.dess-itea.org/deliverables/ITEA-DESS-D181-V01P.pdf

[44] Rational Software Corporation, IBM Rational RequisitePro user's guide, version 2003.06.00., 334 p. (2005-07-26) URL: http://www.se.fh-heilbronn.de/usefulstuff/Rational%20Rose%202003%20Documentation/reqpro_user.pdf

[45] JMR support, Storage Glossary of Terms, (2005-08-26) URL: www.jmr.com/support/glossary.html

[46] Telelogic Synergy CM web page, (2005-09-07) URL: http://www.telelogic.com/products/synergy/cmsynergy/

[47] The Eclipse Platform (2005-09-15) http://www.eclipse.org/

[48] Gurd A. (2004) Dynamic Requirements Management for Iterative/Incremental Development, A Telelogic White Paper, 17 p. (2005-10-25) URL: http://www.telelogic.com/campaigns/2003/global/doors_analyst/papers.cfm

[49] Eclipse Platform Technical Overview, White paper, 20p. (2005-11-03) http://www.eclipse.org/

[50] Cleland-Huang J., Zemont G., and Lukasik W. (2004) A Heterogeneous Solution for Improving the Return on Investment of Requirements Traceability. In: 12th IEEE International Requirements Engineering Conference (RE'04) pp. 230 - 239

10. APPENDICES

Appendix 1 Detailed Tool Analysis
Appendix 2 Use Case Examples for Tools Evaluation

APPENDIX 1. DETAILED TOOL ANALYSIS

CALIBER RM DETAILED ANALYSIS

Multi-platform Support

According to the CaliberRM technical overview, CaliberRM only runs under Windows-based operating systems, so CaliberRM does not support multiple platforms. [22]

Tool Integration

CaliberRM supports integration with following tools:

- Integration with software configuration management tools, such as Borland StarTeam and other products supporting to the Microsoft SCCI API.
- Integration with modelling solutions, such as Borland Together.
- Integration with test management tools, such as Mercury TestDirector.
- Integration with project management tools, such as Microsoft® Project. [23]

Web Access

CaliberRM has two tools that enable customers to manage their requirements using the Internet – *CaliberRM Web* and *CaliberRM WebView*. Each of these tools allows the project team members to connect to a CaliberRM server through a standard browser. Some features are not available through the browser [25]. According to [24], CaliberRM has full support for the Java Applet based browser client.

Access Control

Security is one of the more complex aspects of CaliberRM, due mainly to its flexibility. The user access to every single requirement type, default attribute and custom attribute can be customized. Security profiles can be created and assigned to user groups. The users themselves have to be assigned to one or more custom defined groups. [25]

The security model enforces access from the Project level, Requirements Type level and down to the attribute level of requirements. [24]

Information Sharing

CaliberRM offers a role-based notification tied to the changes of the project requirements. The notification is implemented through automated email notification to all users noted as responsible for a requirement. Notification can be sent to the groups of users for certain requirement changes. [23], [25]

Simultaneous Use

CaliberRM supports multiple concurrent users. The users can utilise a mix of Named User or Concurrent User licenses. [24]

Requirements Identification

Central to CaliberRM is its requirements management capability. It provides users the ability to customize the requirement types and to create custom attributes for those types, such that almost any kind of information can be recorded. At least the following attributes can be provided for all requirements: Name, version, number, hierarchy, owner, status, priority, description, responsibilities, references, traceability, validation, discussion, comment and history. [25]

Prioritization can be done together with the requirements identification. The spreadsheet views of the requirements permit sorting, and prioritizing according to the cost and value is supported. [25]

Req. Classifying and Viewing

Spreadsheet views of requirements permit sorting and prioritizing according to the cost and/or value provided; views can be created on any data collected to make the management of the requirements easier.[22]

Formats

CaliberRM supports MS Word in document importing. MS Word format is used to generate documents by default, but also Excel, HTML, etc. can be generated. [24]

Change Management

CaliberRM includes comprehensive audit trail and change history ensuring that every change is automatically audited. Each change creates a unique history record; differences between one version of a requirement and another are highlighted and the reason for the change recorded. [23]

Traceability

CaliberRM provides two main ways to view traceability relationships. The first is a Traceability diagram that displays the information graphically. The second is a filterable Traceability Grid. [25]

There are essentially three kinds of traceability information that can be recorded in CaliberRM.

- First, there is the inherent traceability provided by the hierarchical nature of requirements. Specifically, there is an implied link between the parent requirements and their sub-requirements.
- The second kind of traceability is one created directly by users between requirements. These are potentially many-to-many, directional traces (from/to) that are assigned through drag-and-drop actions by the user.
- The final kind of traceability is provided via References. These are links created directly by users either to external files, web links, or to internal text files.

Document Importing

Supports wizard driven MS Word document import, which assists in identifying the requirements before actual import is completed. [24]

Document Generation

CaliberRM can produce documentation in a variety of formats, including MS Word. MS Word includes document comparison capabilities and these may in turn be leveraged to facilitate the change/comparison analysis. Output is generated through the Document Factory feature. The Document Factory generates template-based output in MS Word format by default, but can also be used to generate Excel, HTML, etc. [24]

Tailoring and Extensibility

CaliberRM is an company-wide solution for medium to large teams or companies [25]. CaliberRM can easily be customized to support an individual requirements management process, ensuring that the organizations and teams retain control and work the way they are used to.

REQUISITEPRO DETAILED ANALYSIS

Multi-platform support

RequisitePro is MS Windows – based solution; therefore it does not offer multi-platform support. [26]

Tool Integration

RequisitePro integrates with multiple tools in the IBM Software Development Platform to improve accessibility and communication of requirements. [27]

IBM Rational RequisitePro integrates with these other IBM Rational tools:

- Rational ClearQuest for Change request management
- Rational Rose and Rational XDE for Visual modelling
- Rational Unified Process for Processes
- Rational ClearCase for Configuration management
- Rational TestManager for Test management
- An integration with Microsoft Project is also provided

These components are also packaged into Rational Suite, so the integrations between components are supposed to be compatible at each release. [26]

For example, the integration with the TestManager helps to ensure that the requirements serve as a direct input for the test case creation so that the testers and the QA engineers can validate their applications with confidence. As changes to requirements occur, testers can run reports that highlight which test cases are affected by the change, ensuring that their test cases are properly updated to reflect the latest requirements. [28]

Web Access

IBM Rational RequisitePro comes with a Web interface, making requirements accessible to all team members, including remote locations or multi-platform environments. It is possible to access, query, modify, and create requirements, and requirements documents, and manage the traceability through the RequisiteWeb interface. The supported browsers include Microsoft Internet Explorer or Netscape Navigator. [26],[28]

Access Control

User definable groups can have specific security permissions. The security is granular enough to provide access rights at the document level, requirement level, and requirement attribute level, as well as attribute value level. While current security features can restrict the modification of requirements, it is not currently possible to restrict persons from viewing them. [26]

Information Sharing

If requirements change, emails are automatically delivered to notify the predefined stakeholders. [28]

Simultaneous Use

Multiple concurrent users are supported with RequisitePro. [26]

Requirements Identification

The requirements can be organized by the user defined requirement type. Each requirement type defines a visual representation of the requirement text in the documents, as well as a set of user-defined attributes and associated attribute values. RequisitePro provides unlimited user-defined attributes of types: text, list, integer, real, date, time, or URL link for any requirement type. [26]

In accordance with the RequisitePro data sheet, requirements prioritization is supported on some level. [28]

Req. Classifying and Viewing

When requirements are identified, the users can create different types of requirements: performance requirements, system requirements, marketing requirements, test requirements, etc. Each requirement type has its own set of user-defined attributes. Attributes such as weight, cost and risk can be created as needed. [26]

A graphical traceability tree and matrix views show the relationships between the system requirements and the implementation.

Formats

Through integrations with Rational development tools (Software Architect, Software Modeler and Technical Developer), the RequisitePro solution lets the developers connect the requirements to the use-case models, enabling instantaneous access to the use case specifications from the use case diagrams, as well as visibility into the requirements information. [28]

The requirements documents can be formatted using the full MS Word support for documents. A requirements metrics feature allows for plotting of the requirement data in MS Excel. [26]

Change Management

RequisitePro product provides functionality for establishing and analyzing the impact of change. These features allow the user to link the related requirements so that as a change occurs in one requirement, the user can easily see its impact on the other related requirements. The requirement audit trails the document who, what, why and when a requirement modification is made, helping the user to analyze its impact across the project. [26]

The history is provided for every requirement. This includes date, time, author, before and after contents, and annotated change rationale by the author. The history is automatically updated every time a requirement changes. [26]

Again, the integration capability between RequisitePro and Rational Software Architect and Software Modeler can enable the users to connect the design elements

to the requirements. By linking the design elements with the requirements, you can easily review and assess the impact of the requirements changes on the design elements and keep the developers informed of changes that may affect their work. [26],[28]

Traceability

RequisitePro provides detailed traceability views that display the parent/child relationships and show requirements that may be affected by upstream or downstream change. [27]

RequisitePro's traceability capability provides a query mechanism to efficiently detect the traceability links. It also allows flexible queries to find linked and unlinked requirements. The traceability trees are provided to view the complete path of the relationships. A "Requirement Go To" feature aids navigation. The querying of the traceability relationships verifies the requirement coverage. The requirements attributes can be used to track, who created the requirement, who implemented it, and who tested it. [26]

Document Importing

RequisitePro provides an import wizard to import requirements either from a Word document or a Comma-Separated Value (CSV) file. Word documents can be parsed for the requirement extraction based on user-defined keywords, Word headings (including Word highlight), and text delimiters. CSV files provide the ability to import requirement attributes as well. The import facility is also available for a block of text in a document. System implementation documents can also be imported into RequisitePro. Any appropriate text can then be made into system implementation requirements. These system implementation requirements can then be traced to system requirements. [26]

Document Generation

Through the integration with IBM Rational SoDA (a reporting tool), RequisitePro supports the generation of documents in any format for which there is a Word template. Requirement documents, written in Word, are managed in-situ by RequisitePro, and can be formatted using any MS Word functions or templates. Reports from the requirement repository can also be extracted directly into a formatted Word document. [26]

Tailoring and Extensibility

RequisitePro is a solution for small to medium teams and companies.

DOORS/ERS DETAILED ANALYSIS

Multi-platform Support

DOORS can be run under Microsoft Windows NT 4, 2000, XP, 2003 server, Sun Solaris 7 and 8HP/UX 11, so multi-platform criterion is supported. [29]

Tool Integration

DOORS has over 25 interfaces to the most popular tools for design, analysis, text, CM, etc. [29] For a full list of current interfaces contact a Telelogic representative or visit [30].

DOORS has flexible method of inter-tool communication. There is a full API and the DOORS extension language (DXL), which can be used to write imports and exports to other tools in many formats. DOORS on the PC can also use OLE automation for integration, such as those used by Microsoft tools.

DOORS integrates with Telelogic TAU, Telelogic SYNERGY and Telelogic DocExpress. [31]

Web Access

DOORS supports web access with DOORSnet, which can be used as an interface to the central requirements repository through a web browser [29]. Accessed via the web, DOORSnet enables users to access live DOORS data remotely. They can review and edit the data without needing to install DOORS. [31]

Access Control

Access control may be achieved on three levels in DOORS. First, sets of data, such as requirements in a document or all test cases, may be controlled. Second, the individual objects, such as a single requirement, may be controlled. Third, the access controls may be imposed on attributes within objects. For example, users may be able to edit a comments attribute but not modify the allocated cost attribute. Access rights can also be inherited by children from parent objects.

The access levels include the ability to read, create, modify, delete and control accesses itself. Further, a mechanism called propagation allows access right to be imposed on documents or requirements not yet in existence.

Access by the team to the database must be tempered by multi-level access control (i.e. the ability to protect things from being modified). This also includes the ability to submit changes into an approval cycle (for acceptance/voting) before committing the changes to the tool for everyone to see. [29]

Information Sharing

DOORS' proactive Suspect Links provides an automatic change notification and a detailed impact assessment so the impact of the changes can be understood. [31]

Simultaneous Use

DOORS provides thousands of concurrent users with a single, customizable view of the most up-to-date requirements information.

- First, DOORS supports multi-user concurrent write access to the same document.
- Second, DOORS' CPS (Change Proposal System) allows proposed changes from multiple users to be reviewed together and either the best taken or a combination generated. This feature is also available through the DOORS web interface, DOORSnet.
- Third, DOORS Distributed Data Management (DDM) allows portions of the DOORS database to be taken out, worked on, and returned for resynchronization with the database.
- Fourth, DOORSrequireIT can be used to extract a document from DOORS into Word. Here, the requirements and their attributes can be managed, modified, deleted, and new ones created before returning it to the DOORS database for an update.
- Fifth, DOORSnet allows users from remote locations also to participate in the teamwork by making changes and creating new requirements directly to the DOORS database using the internet or an intranet. [29]

Requirements Identification

With DOORS the users can define attributes for additional requirement information, such as owner, priority, cost, risk, etc. [31]

Req. Classifying and Viewing

Users can define views, thus maximizing efficiency for different user categories. Views allow you to display only the data that you need for your current work. With DOORS, the user can define attributes for additional requirement information, such as like owner, priority, cost, risk, etc. [31]

The DOORS document-style interface provides thousands of concurrent users with a single, customizable view of the most up-to-date requirements information. [29]

Formats

DOORS/Analyst provides a mechanism of describing functional decomposition and analysis in the form of UML 2, stored, viewed and edited directly within DOORS. [29]

DOORS supports most of the popular formats used for importing and exporting including MS Word, RTF, Interleaf, FrameMaker, text and spreadsheets. [31]

Change Management

DOORS automatically records who, what (down to the actual attribute changed), when and how, automatically. The why can also be captured either through the voluntary entry of data by the user, or forced via DOORS' automated trigger

mechanism, such that the user would not be able to save the change without entering a rationale.

DOORS also supports a full Change Proposal System (CPS or ECPS) for collecting change requests and formally reviewing them via a CCB before changes get into the documents or data sets. The CPS is also available in DOORSNet for Internet/Intranet access. [29]

DOORS integrates with a variety of tools to enable users to perform impact analysis or to verify compliancy to requirements throughout the lifecycle. Data can be linked and generated into clear documentation that shows full traceability from requirements to design, testing and deployment. Consequently, the impact of requirements changes can be seen and understood at all times. [31]

Traceability

DOORS document hierarchies may be viewed graphically and traceability may be viewed as “tree” structures in the Traceability Explorer. The linking of requirements to the system elements can be accomplished from either end of the link – from the implementation back to the requirement or from the requirement down to the system element. This is the natural way of working in DOORS and very little difference is made between linking up, down or on the same level. [29]

DOORS provides a complete audit trail by enabling you to link related data and multiple documents and access it in a single, up-to-date view. Users can utilize link matrices or, most simply, drag and drop between items.

DOORS integrates with a variety of tools to enable users to perform impact analysis or to verify compliancy to requirements throughout the lifecycle. Data can be linked and generated into clear documentation that shows full traceability from requirements to design, testing and deployment. Consequently, the impact of requirements changes can be seen and understood at all times. [31]

Document Importing

DOORS can import information in many forms such as MS-Word, ASCII, Spreadsheet, FrameMaker, Interleaf and RTF, so that structures, attributes and links may be set up automatically without manual input. [29]

DOORS uses the popular formats for importing and exporting, including MS Word, RTF, Interleaf, FrameMaker, text and spreadsheets. [31]

Document Generation

DOORS can generate colour graphs and charts for displaying metrics data, results of calculations and statistics. Examples of such charts produced by DOORS include a volatility chart showing numbers of changes over time for a document or a data set. These charts and graphs can be generated and printed directly from DOORS without the use of an external graphics package. DOORS document hierarchies may be viewed graphically and traceability may be viewed as “tree” structures in the Traceability Explorer. [29]

Tailoring and Extensibility

DOORS offers multi-platform support, so companies can easily introduce it in their system environments. It also has quite broad integrity support to the other tools. DOORS is an company-wide solution.

APPENDIX 2. USE CASE EXAMPLES FOR TOOLS EVALUATION

Table 5. Use case #1

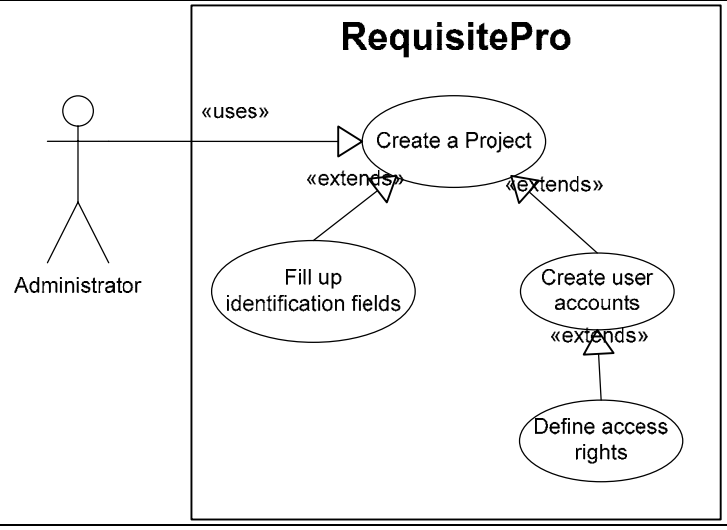
Use case:	Create a Project
Summary:	Administrator creates a project and defines user accounts and access rights for all project members.
Frequent:	Once when the project begins. Users can be added later if new project members join in the project.
Purpose:	Create a project, where requirements management issues are handled during the project. In addition, user accounts and access rights for them are defined, so that different users have appropriate access rights to the requirement data.
Precondition:	RequisitePro is installed and configured and it works well.
Description:	Administrator creates a project to the requirements database in central repository. Identification fields, such as project name, database path, etc. are defined, and thereby every project can be distinguished from each other. When a project is created, the administrator creates user accounts and access rights for every user. This makes it certain that the user can edit the change requirements only if it is necessary. Access rights can be, for example, role-based or project-based.
Figure:	

Table 6. Use case #2

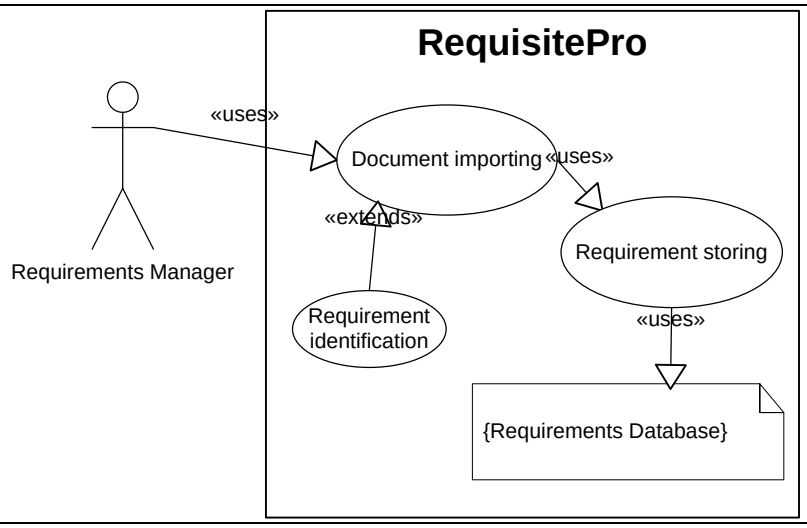
Use case:	Document importing
Summary:	Existing requirements specification document is imported to the RequisitePro.
Frequent:	Once in the beginning of the project.
Purpose:	To import an existing requirements specification document.
Precondition:	Requirements specification exists and project is running.
Description:	Requirements Manager imports the existing requirements specification document to the RequisitePro by using its document-importing feature. In order for RequisitePro to be able to recognize requirements from the document, they have to be identified by using certain identification methods. In practise, every requirement must be chosen by marking them manually in the document, or they can be identified by using a certain identification tag that is repeated in every requirement (e.g. REQ1, REQ2, then REQ can be the identification tag). RequisitePro automatically adds these identified requirements to the database.
Figure:	

Table 7. Use case #3

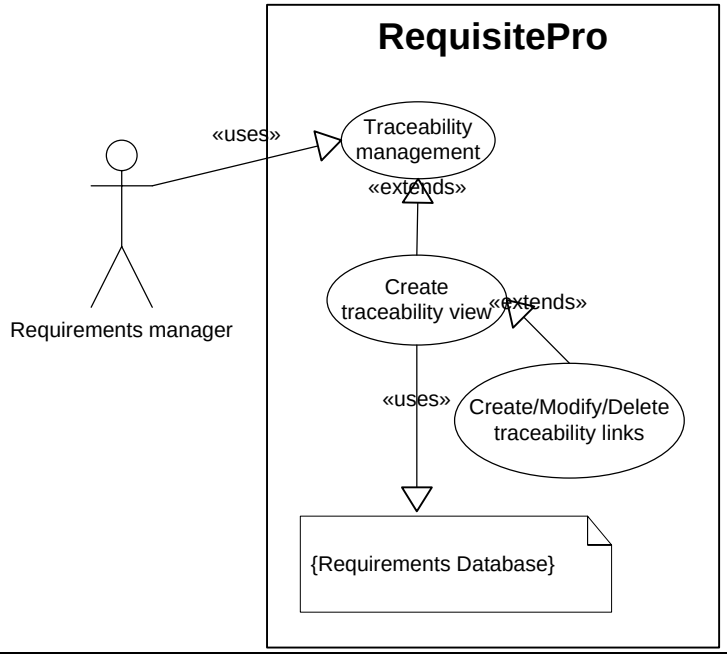
Use case:	Requirements traceability management
Summary:	Requirement is traced, which means all its links and relations to the other requirements are clarified. (SR – FEAT, FEAT – STRQ, SR – SR etc.)
Frequent:	Always when a requirement is created, later if necessary.
Purpose:	To track relationships between all requirements, thereby making it easier to take care of the change impact assessment.
Precondition:	The requirements, that are being traced, exist.
Description:	The requirements manager adds traceability links between SW requirements and product features. The traceability links can also be added between product features and stakeholder requests, and use case requirements.
Figure:	

Table 8. Use case #4

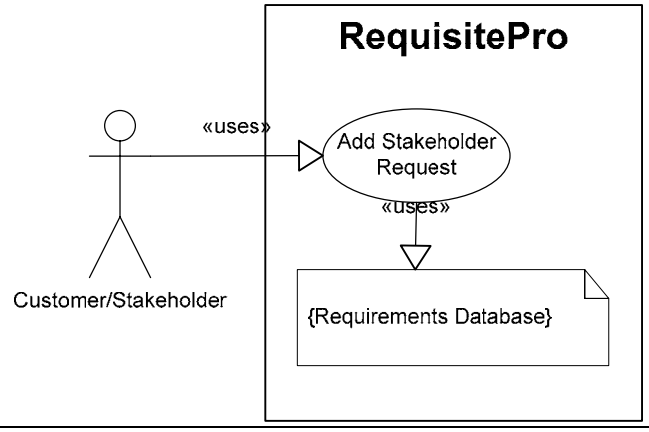
Use case:	Add stakeholder request.
Summary:	For some reason, the stakeholder wants to add a stakeholder request pertaining to a certain requirement using a web access of RequisitePro named RequisiteWeb.
Frequent:	Once in a while, when necessary.
Purpose:	The customer is not satisfied in some requirement and he/she wants to change it or add a new requirement.
Precondition:	The customer has web access, user account and appropriate access rights to the system.
Description:	The customer wants to change one requirement. He/she logs in to the system by using web access and uses the requirements finding feature. After that, he/she fills the change request form with adequate information (who, what, when, etc.) and links it to the desired requirement. Then the stakeholder request is sent to the system when the system informs the user that the request is received.
Figure:	

Table 9. Use case #5

Use case:	Change impact assessment.
Summary:	Assessment of change impact to the project and product.
Frequent:	Always when making a change decision.
Purpose:	The change impact assessment helps making change decisions.
Precondition:	The customer has made the change request and the appropriate person has preliminarily accepted it.
Description:	When a requirement is changed, the user can open the traceability view or change the impact assessment view and see where the change influences. The change makes traceability links to become "suspect". Then the user can follow these links and check whether it is necessary to change these requirements, too. If the requirements are linked to the project management artefacts, the user can easily check what project tasks must be re-planned. RequisitePro provides impact analysis views to see where, for example, a changed stakeholder request or a software requirement impacts. Every change affects the project course (time, efforts, etc.) to some extent and, of course, the product itself. If the requirements are linked to the project management artefacts, the user can easily check what project tasks must be re-planned.
Figure: