Identification of Improvement Issues Using a Lightweight Triangulation Approach

Tony Gorschek (tgo@bth.se) and Claes Wohlin (*claes.wohlin@bth.se*)
Department of Software Engineering and Computer Science,
Blekinge Institute of Technology, PO Box 520, SE-372 25 Ronneby, Sweden
Phone: +46 457 385000

Abstract
One of the challenges in requirements engineering is the ability to improve the process and establish one that is "good-enough". The objective of this paper is to present a lightweight approach to identify process improvement issues. The approach is developed to capture both the views of different stakeholders and different sources of information. An industrial investigation from a small company is presented. In the investigation both projects and the line organization have been interviewed and documentation from them has been studied to capture key issues for improvement. The issues identified from one source are checked against other sources. The dependencies between the issues have been studied. In total nine issues for improvement of the requirements engineering work at the company were identified. It is concluded that the approach is effective in capturing issues, and that the approach helps different stakeholders to get their view represented in the process improvement work.

Keywords

Requirements Engineering, Software Process Improvement, Data Triangulation, Process Evaluation.

1 Introduction

The area of requirements engineering (RE) is often underestimated in value in the area of software engineering, in spite of the fact that requirements problems are the largest cause in project failure. [1, 2].

This paper presents an investigation conducted in industry aimed at introducing a way to identify RE *improvement issues* in small and medium sized enterprises (SME) [3]. The lightweight approach presented uses data point *triangulation* [4] as a means to identify possible improvement issues. The data points consist of two major parts, i.e. a *line study*, and a *project study*. This is depicted in Figure 1. The project study is comprised of three exploratory case studies [4], and each case study is in turn comprised of project specific interviews and documentation (see section 3.2.).

The line study is comprised of line interviews and documentation (see section 3.3.). The main objective of the investigation was to ascertain state-of-practice at Danaher Motion Särö (DHR) (see section 2.) and to identify improvement points in their RE process. There are approaches for RE process improvement targeted at e.g. SME. Sommerville and Sawyer [5] for instance present a series of guidelines as well as a guide for process improvement as a product of the REAIMS project [6]. The PERE (Process Evaluation in Requirements Engineering) method used for analysis of the requirements engineering process from two different viewpoints in REAIMS is basically a systematic approach to understanding processes, and a human factors analysis of the process. The lightweight approach presented in this paper uses data point triangulation with *four* main data sources (see section 3.1), In addition to this the emphasis of the approach presented in this paper is on fast and fairly low-cost evaluation.

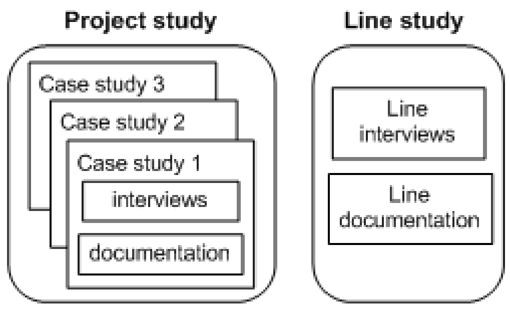

Figure 1. Investigation description.

In addition to RE specific improvement strategies there are some more general, targeted at quality assurance in software projects, such as *The TickIT Guide* (ISO 9001:2000) [7]. These process improvement and quality assurance approaches do not offer any lightweight identification of how to establish what needs to be improved in an RE process, i.e. how to identify improvement issues.

The main objectives (contributions) of the paper is to present (i) the lightweight triangulation approach used for the identification of the improvement issues (presented in section 3), (ii) the improvement points identified and triangulated (presented in section 4), and (iii) the dependencies between the improvement points (also presented in section 4). In addition to this there is an overview of state-of-the-art in relation to each improvement point (see section 5). Section 6 contains the conclusions drawn in the paper.

2 Investigation Context

Danaher Motion Särö develops and sells software and hardware equipment for navigation, control, fleet management and service for Automated Guided Vehicle (AGV) systems. More than 50 AGV system suppliers worldwide are using DHR technologies and know-how together with their own products in effective transport and logistic solutions to various markets worldwide. The headquarters and R & D Centre is located in Särö, south of Gothenburg, Sweden. DHR has approximately 100 employees. DHR is certified according to SS-EN ISO 9001:1994, but is uncertified according to CMM and CMMI.

DHR has a wide product portfolio, as the ability to offer partners and customers a wide selection of general variants of hardware and supporting software is regarded as important. Tailoring and especially lighter customer adaptation often follows the procurement and subsequent installation of a system.

This in addition to development of new software and hardware makes it a necessity to plan, execute and manage a wide range of projects.

Requirements are one factor that binds all of the projects together. It is not necessarily so that requirements breach project boundaries (although this is known to happen) but rather that most projects involve elicitation, analysis and negotiation, management and documentation of requirements. In addition to this the stakeholders (or groups of stakeholders) are at least as divergent and diversified as the projects themselves. Requirements from general sources like *the market* have to be taken into consideration as well as the ones from *industry-partners*, *end-customers* and *internal sources* such as developers and management. All of these factors contribute to a need for an RE process that is good enough (at present) and has the potential to meet growing demands and complexity (in the future). The ability to prepare for the future, and improve current practices, is the underlying motivator for the work conducted in the partnership between DHR and Blekinge Institute of Technology.

The first step in this partnership was to map the RE process at DHR (establish a baseline), and to determine what the main improvements issues were.

3 Investigation Design

In this section the overall design of our *multi method investigation* is described, how the data was elicited for each data source (project study and line study), compiled and analyzed. A validity evaluation is also presented below.

3.1 Multi Method

By looking at several data sources instead of relying on a single one, e.g. solely a case study, a higher level of validity can be achieved [8]. Below four major data sources are described, (A) data from the case study interviews, (B) data from project documentation, (C) data from line interviews, and (D) data gathered from line documentation. (A) and (B) together comprise the project study, i.e. interviews and project documentation from three specific projects (see section 3.2.). (C) and (D) are interviews and documentation from the line, and together comprise the line study. The idea is not to use all of the data sources solely for the purpose of getting more data, but rather to have a confirmation (validation) of the individual issues identified. This is achieved through triangulation, illustrated in Figure 2. One issue is identified and specified, then checked against the other data sources for confirmation. An additional benefit is that this usage of different sources enables several perspectives, decreasing the possibility of missing crucial information.

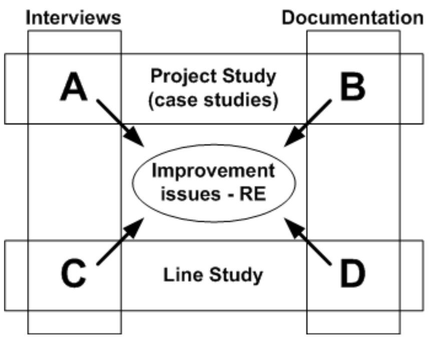

Figure 2. Triangulation using several data sources.

In addition to getting project specific data (A) and (B), data is elicited from the line (C) and (D), i.e. the development support and production parts of the organization (see section 3.3.).

In addition to the horizontal division of the data collected there is a vertical distinction to be made. Data gathered from the interviews, (A) and (C), are complemented and/or verified (occasionally contradicted) by the data gathered from the documentation, (B) and (D). When conducting the triangulation two leading data sources were selected, namely the interviews (A) conducted in the case studies and the interviews conducted in the line study (C). The reason for selecting leading data sources was the need for points of reference that could be compared (triangulated). If no leading source was selected every single potential data point had to be considered and triangulated. This would have been possible but very time and resource consuming, and the final improvement issues identified would be the same.

It was however not a foregone conclusion that specifically the interviews were to be chosen as the leading data sources. The main reason for the decision was the fact that the interviews reflected the views of the project and line team members. In addition to this a piece of documentation (whether it was from the project study or the line study) was primarily used for confirmation of the issues identified during the interviews. If the documentation was used for the identification of issues themselves all things not present in the documents could be considered potential issues. This became a problem if the contents of the documents were compared to state-of-the-art, in which case the absence of things in DHR's documentation could yield any number of issues depending on what sources in state-of-the-art were used for comparison. The idea was not to identify all things not performed and turn each these in to an issue (basically getting a maximum sized RE process), but rather to identify what needed to be changed/added to DHR's RE process based on their own perception and experiences.

Three other reasons existed for making the interviews the leading data source. The case study interviews were the source of pre-verified quantitative data (the pre-verification is described in section 3.2.1.), the case studies were the source of the most up-to-date data (three current projects were studied), and last the case studies yielded the lion share of the total data collected.

3.2 Project Study (Case Study) Design

In order to get an overview of the RE process no single case could be studied that was representative for most of the projects conducted at DHR, rather a block of projects had to be identified [9]. Several projects had to be considered and three were finally chosen. The main criteria were getting a cross-section of typical projects conducted, and to get a wide representation of the developers and managers involved in the projects, i.e. avoid asking the same people questions about different projects. Company representatives with an overview of DHR's domain, project portfolio and current practices picked three projects. The projects chosen were of three types:

- ***Project Alpha*** Externally initiated software development project aimed at an end customer. Basically an addition of customer specific features to an existing piece of software.
- ***Project Beta*** Externally initiated software development project aimed at an industry partner. Basically development of a new service tool to be used by an industrial partner in their work with end-customers.
- ***Project Gamma*** Internally initiated software and hardware project aimed at updating a certain central product. Basically a generational shift to newer technology for mainly production aspects.

The division of the projects can be seen as two steps, first there is the obvious distinction between the projects from the point of what sort of product they yield, bespoke (alpha and beta) or generic (gamma). As described by Sommerville [10] bespoke products are aimed at specific customers and the projects are initiated by external sources, generic products however are rather aimed at a market and initiated internally by the producer. Second there was a need for a further granulation of the "bespoke project type". This is due to the fact that project alpha was aimed at an end-customer, while beta was aimed at an industrial partner. The preconditions of a partner project differ from that of an end-customer. Industrial partners often have more domain specific and technical knowledge.

Subsequent to project division and selection, interviews were booked with representatives for all *roles* in the projects at hand. Four major roles were identified that were typical for all of the projects in question, *Orderer*, *Project Manager*, *System Engineer* and *Developer*. These official roles can briefly be described in the following manner:

1. The *Orderer* has the task of being the internal owner of a certain project, i.e. has the customer role and if applicable the official contact with an external customer and/or partner. This party is responsible for the official signing-off when it comes to the requirements, i.e. he places an order.
2. The *Project Manager* has the traditional role of managing the project, resources, planning and follow-up. As far as requirements are concerned the Project Manager is responsible for that the requirements engineering is performed, the requirements specification is written and signed off by the System Engineer.

3. The *System Engineer* is the technical responsible for a project. It is also important to recognize that the System Engineer has the official responsibility for the requirements specification in a project.
4. The *Developer* is a representative for the developers (e.g. programmers) in a project, the ones actually implementing the requirements. The developers use the requirements specification.

The researchers in cooperation with representatives for DHR identified these roles. In total 11 people were interviewed during the case studies, four each from projects Alpha and Gamma, but only three from project Beta. The Orderer from project Beta was not accessible at the time of the investigation.

3.2.1 Case Study Interview Questions

Semi-structured interviews [4] were used for the case studies. There were 31 questions in total, two warm-up questions where the subjects stated name, title and responsibilities etc. Following warm-up the questions were divided into four main parts, i.e. *requirements elicitation*, *requirements analysis and negotiation*, *requirements management* and a *general part*. The questions posed in the first three parts were aimed at finding out what tasks were performed in the context of RE for the project in question, who performed them, when and how. An example of such a questions could be “was there any classification of the requirements into classes, groups or types during the elicitation process”, and the follow-up questions “how and by whom was this done”, and “what were the criteria for the classification…”. The subjects were asked to elaborate on their answers, especially if the answer was a simple “no”. The intention is to get an idea of why a certain task was not performed. The structure of the questionnaire and the tasks that the questions were based upon were inspired primarily by the work of Sommerville & Sawyer [5] and Gorschek et al. [11]. The questions posed in the first three parts were of a qualitative nature.

The general part (fourth) contained three questions (from a total of four) of a quantitative nature. Each of these questions was comprised of several sub-questions, and two of them were *open*, i.e. up to the subject to come up with the contents and not merely answer if a certain task was performed. An example of such a question was “list three things that work best with respect to RE at you company” and the follow-up “…three things that have the best improvement potential with respect to RE…”. These open questions were beneficial from at least three aspects, i.e. the data gathered during preceding questions could substantiate the answers given here, items missed by earlier questions could be observed, and last the subjects were asked for their opinion without being led in a certain direction.

When the interviews were completed the results were compiled and the quantitative results were used as a primary source of data, and the qualitative results were used as a secondary source as well as to validate the quantitative data. Figure 3 shows the procedure, the reasoning behind this was to be able to see patterns in the quantitative data that could be substantiated (validation) and augmented by the data gathered from the qualitative part. The confirmed data was then used as one of the sources (A) in the triangulation (see Figure 2).

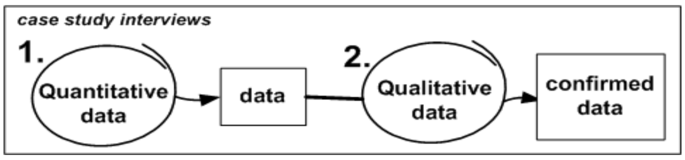

Figure 3. Pre-validation of case study data.

The subjects were given the questions on location. Each interview took about one to one and a half hours. All of the interviews were conducted over a period of five consecutive days. In addition to general information given to the subjects they were assured that all answers would be anonymous. The questionnaire used can be obtained through the authors. A digital recorder was used during the interviews in addition to the notes taken, this to enable further clarification and analysis after the fact.

3.2.2 Project Documentation

The study of documentation was a substantial part of the overall investigation. The documentation in question was of two types, first there were documents pertaining to the projects investigated in the case studies (B), and second there was line documentation (D) (see Figure 2).

Project documentation included pre-study documents, project specifications, project plans, requirements specification, subsequent follow-up documentation, budget and time reports and meeting minutes.

3.3 Line Study

In addition to the 11 case study interviews conducted three line interviews were performed. This was done to elicit data from three additional roles not present in any of the block projects. The roles interviewed consisted of representatives for *System Test*, *Production* and *Application Developer.* The descriptions of the roles below are according to the official line documentation.

- The System Test role can be described as the traditional role of application and feature test. This role is officially present during initial project meetings and is a part of the verification of the requirements specification.
- The Production role also has a presence in projects. This representation consists of verifying that the productional aspects are met, i.e. that production is taken into consideration at an early stage.
- The Application Developer role represents installation and adaptation aimed at industry partners and/or end customers. Adaptation here translates to tailoring, development of some light customer features and some support.

The interviews conducted here were unstructured [4], i.e. some directional questions were used, but for the most part informal conversation dominated. The conversational interview's goal was to elicit data about the lines role in the RE process, the representatives opinions and experience. The structure observed during the case study interviews (see section 3.2.1) was present to a certain degree, i.e. the interviewer wanted answers to some of the same questions, but no questionnaire was formally used. The reason for using a more unstructured approach during the line interviews was due to that they were somewhat more general in nature, i.e. did not refer to a specific project. The same general information and assurance of anonymity as before was issued.

3.3.1 Line Documentation

Line documentation (D) (see Figure 2) included general process descriptions, development process descriptions, process improvement strategies, documentation pertaining to roles and responsibilities, and the requirements engineering process description. Generally one could say that the line documentation was a collection of documents in which the official processes and responsibilities were specified.

3.4 Validity Evaluation

In this section we discuss the threats to this investigation. We base this on the discussion of validity and threats to research projects presented in Wohlin et al. [12]. One type of threats mentioned in [12] is not relevant, since the investigation is conducted in an industrial environment. The threat not considered is construct validity, which mainly is concerned with the mapping from the real world to the laboratory. The investigation presented here is however conducted in the real world. The validity threats considered are: conclusion, internal and external validity threats respectively.

3.4.1 Conclusion validity

The questionnaire used for the structured interviews was validated through preliminary testing and proofreading by several independent parties, this to avoid factors like poor question wording and erroneous formulation.

Each case study interview was conducted without any break. Thus the answers were not influenced by internal discussions about the questions during e.g. coffee breaks.

The sampling technique used for the case studies, i.e. projects selected and interview subjects for every project, can pose a threat to the validity of the investigation. The projects may not be totally representative for the projects conducted at DHR, in a similar manner the interview subjects may also not be representative for the role they represent. Three things alleviate these potential threats. First the fact that three cases was studied, and this also gives us three representatives interviewed for each role identified. The exception to this is the Orderer in project B. Third is the triangulation effect, i.e. data from different data sources is used for validation.

3.4.2 Internal Validity

The use of a digital recorder during the interviews could be considered as a problem due to the fact that certain people may feel constrained to answer differently if an interview is recorded. This potential problem was alleviated by the guarantee of anonymity as to all information divulged during the interview, and that the recordings are only to be used by the researchers.

3.4.3 External Validity

The external validity is concerned with the ability to generalize the results. This is not a main threat in the investigation, since the objective is not to generalize the improvement issues to another environment. The important generalization here is whether the applied approach for identifying improvement issues is possible to apply in other environments. There is nothing in the approach that makes it tailored to the specific setting ad hence the approach should be useful at other small and medium sized enterprises that would like to identify improvement issues in the requirements engineering process.

4 Results

4.1 Project Study

4.1.1 Case Study Interviews (A)

Data from the case study interviews resulted in nine general improvements issues summarized in Table 1. In the first column there is a unique id for each improvement issue, the second column holds the name of each issue, and the last column houses the support number of each issue. The support number is the total number of times an issue was brought up during the interviews, i.e. issue 1 has a support number of 5, meaning that five out of eleven people interviewed brought up this issue during the open questions (see section 3.2.1.).

Issue Id	Improvement Issue	Support number
1	Abstraction level & Contents of requirements	5
2	Requirements overview in projects & over project boundaries	4
3	Requirements prioritization	4
4	Requirements upkeep during & post project	4
5	Requirements responsible/owner with overview of the requirements	3
6	Roles and responsibilities RE process	2
7	System test performed against requirements	2
8	Customer relations during RE	2
9	RE process/methods	3

Table 1. Improvement issues case study.

Issue-1: A central issue for improvement is how the requirements are specified, contents of each requirement and to establish a detailed enough level of description. This relates to usability and understandability of the requirements (how well a specified requirement depicts what it is intended to), and to comparability between requirements.

Issue-2: Requirements overview in projects and over project boundaries is an issue viewed as important. In order to facilitate this overview three main other issues have to be satisfied, i.e. Issue-1, Issue-4 and Issue-5. Figure 4 illustrates this, where the relations between the nine issues can be viewed. Issue-2 is *dependent on* Issue-1, Issue-4 and Issue-5 (denoted by the arrows). The relations in Figure 4 can be several levels deep, e.g. Issue-2 is dependent on Issue-4 which in turn is dependent on Issue-5 and so on. Below only the direct relations are addressed during the description of each issue.

Issue-3: In order to meet the demands posed on projects by resource and time limitations, requirements should be prioritized. The level of prioritization should reflect the need on a project basis. Prerequisites for this issue are that the requirements are specified in an adequate way and at a comparable level of abstraction (Issue-1), and that there is an overview of the requirements in question (Issue-2).

Issue-4: The requirements (requirements specification) should be kept up-to-date during and post project. The point of keeping the requirements 'alive' during the project, and not discard them after a certain point in the project, was made.

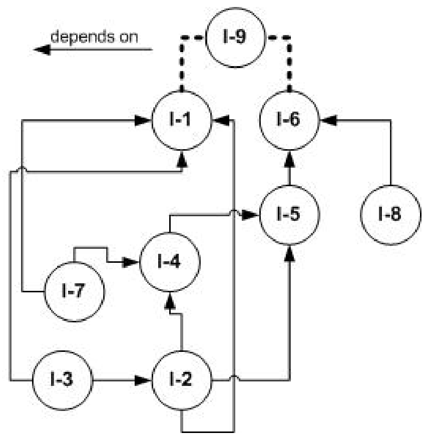

Figure 4. Dependency diagram.

Issue-5: There should be a person(s) responsible for the RE as a whole that has an overview of the requirements. This role should further be the owner of the requirements, i.e. be up-to-date on the requirements and have executive powers and responsibilities pertaining to change to and addition of new requirements. This issue has an obvious relation to Issue-6.

Issue-6: The roles and responsibilities of project members and other stakeholders, e.g. customer, should be clearly and unambiguously defined pertaining to RE.

Issue-7: System tests should be performed against the requirements. Prerequisites for this are that the requirements specification is kept up to date during the project (and post project in some cases) (Issue-4), and that the requirements are specified adequately and at a specified level of abstraction.

Issue-8: The relations and particularly the flow of information between the customer (a group name used for all executive stakeholders, i.e. stakeholders representing the customer in a formal and executive capacity [11]) and the project team vary with project. There is a general view that the information from the customer is passed through too many filters before it reaches the project team. In addition to this the customer is often not accessible (at least not directly) for elicitation and clarification purposes during a project. A prerequisite identified here for this issue is that roles and responsibilities during RE are defined (Issue-6) (see Figure 4), i.e. all persons involved in the RE process (including the customers) are stakeholders, thus they are a part of the responsibility paradigm as well as the members of a project team.

Issue-9: This issue refers to the view that there is a lack of a formalized and/or structured RE process, or methods supporting certain tasks performed during RE at DHR. Issue-9 is to some extent outside of the dependency diagram (see Figure 4) as far as the level of abstraction. All issues except Issue-9 can be described as more or less specific tasks to be performed or rules to be upheld, while Issue-9 rather is an 'umbrella' issue under which the rest can be sorted. Many of the other issues contributes to Issue-9, this is illustrated in Figure 4.

4.1.2 Case Study Project Documentation (B)

Looking at the project documentation in the case study projects several of the improvement issues described in section 4.1.1. are substantiated. Table 2 shows each issue (sorted by issue id), the cells in the B-column (B for triangulation point B, see Figure 2) with a ticked box denote the substantiation, and the note-column offers a short account for it. Four issues could not be directly substantiated by the project documentation, i.e. Issue-2, Issue-5, Issue-6 and Issue-8.

4.2 Line Study

4.2.1 Line Study Interviews (C)

Looking at the line interviews several of the issues described in 4.1.1. are substantiated (denoted by the ticked boxes in column C of Table 3, C standing for triangulation point C, see Figure 2). Four issues could not be directly substantiated by the line interviews, i.e. Issue-2, Issue-3, Issue-5 and Issue-8. It is important to notice that three new improvement issues are identified during the line interviews, i.e. Issue-10 to Issue-12.

Issue-10: There are no mechanisms or practices implemented for requirements reuse. There is reuse on the design/code/function/solution levels however.

Issue-11: No traceability policies are implemented for requirements. In some cases it is possible to trace requirements to the initiating party (backward-from traceability) [5, 11], and to partially trace requirements through design documents to implemented components (forward-from traceability) [5, 11]. These traceability possibilities are however dependent on individual knowledge by project team members and they are not documented.

B	Issue Id	Note
✓	1	The abstraction level and contents of the requirements tend to vary. This is seen within projects, and over project boundaries, i.e. requirements are not comparable.
	2	
✓	3	No requirements prioritization is documented, and no priority grouping can be observed.
✓	4	The update of the requirements specifications (RS) varies, however no RS was ever updated after the first 50% of the project calendar-time had elapsed.
	5	
	6	
✓	7	The problems with Issue-1 and Issue-4 make system test based on requirements difficult.
	8	
✓	9	The RE process/methods seems not to be good enough. This is based primarily on issues substantiated above, but the fact that the level of RE and the methods used varies with project, also indicates this.

Table 2. Improvement issues documentation.

C	Issue Id	Note
✓	1	Abstraction level & Contents of requirements varies and may be inadequate for tasks such as system test.
	2	
	3	
✓	4	Outdated req. specifications make it difficult to use the req. document.
	5	
✓	6	The roles of the interviewed as to their roles and responsibilities in RE are not clear.
✓	7	System test are seldom performed against requirements due to Issue-1 and Issue-4.
	8	
✓	9	See Issue-1, Issue-2, Issue-4 to Issue-8
Improvement Issues from line interviews		
NEW	10	Requirements reuse
NEW	11	Requirements traceability
NEW	12	Requirements version handling

Table 3. Improvement issues line interviews.

Issue-12: There is no version handling of individual requirements. The requirements document itself is however updated and released with different versions.

4.2.2 Line Study Documentation (D)

Several issues identified thus far (see section 4.1.1. and 4.2.1.) were not substantiated by the line documentation, i.e. Issue-2 and Issue-4 to Issue-8. Table 4 offers an overview of this.

D	Issue Id	note
✓	1	The template used for specifying requirements is fairly abstract, i.e. on a high level of abstraction. There are no detailed instructions and/or no detailed examples.
	2	
✓	3	No instructions/help/regulations/method description exists to aid in the prioritization of requirements.
	4	
	5	
	6	
	7	
	8	
✓	9	see Issue-1, Issue-3
Improvement Issues from line interviews		
✓	10	No policies for requirements reuse exist.
✓	11	No policies for requirements traceability exist.
✓	12	No policies for requirements version handling exist.

Table 4. Improvement issues line documentation.

4.3 Triangulation of Results

Table 5 offers an overview of all issues presented, and information about substantiation from the different data sources, i.e. A to D. Nine out of a total of twelve issues were substantiated by two or more data sources (see section 3.1.), and are considered to be triangulated.

Issues 2, 5 and 8 each have a total triangulation value of one, i.e. the proof for the issues could not be distinctively identified in the project documentation (B), were not mentioned during the line interviews (C), and could not be found in the line documentation (D). Thus, these are not viewed as being triangulated.

Issue Id	A	B	C	D	TOTAL
1	✓	✓	✓	✓	*4*
2	✓				*1*
3	✓	✓		✓	*3*
4	✓	✓	✓		*3*
5	✓				*1*
6	✓		✓		*2*
7	✓	✓	✓		*3*
8	✓				*1*
9	✓	✓	✓	✓	*4*
10	▨	▨	✓	✓	*2*
11	▨	▨	✓	✓	*2*
12	▨	▨	✓	✓	*2*

Table 5. Triangulation of improvement issues.

5 Relation to State-of-the-art

In this section each triangulated issue (see section 4.3.) is posted with a description of how some sources in state-of-the-art can contribute in resolving them.

Issue-1: *Abstraction level & Contents of requirements*. Requirements should be stated in a clear and unambiguous way. Although there are many textbooks describing how this is done many specifications are inadequate [1, 13, 5, 14]. Requirements are in addition totally dependent on the customer, i.e. the stakeholders making the demands. Here it is assumed that the elicitation process and stakeholder consulting [5, 11, 15] is not a factor, i.e. all information is available and understood by the party writing the specification. This is however seldom the case, i.e. lack of domain knowledge and miscommunication between developers and customers is often a factor influencing the quality of a requirement [16].

Natural language (NL) specifications are commonly used and there several ways in which to approach this, spanning from the psychology behind getting the right requirements on paper [17], to how important linguistics [18] and ethnography [19] are in the context.

Formal specifications [20] are an alternative (or a complement) to NL specifications. Either way several other techniques can be used in combination with both, e.g. prototypes, scenarios, state diagrams, use cases, data-flow diagrams and so on, to validate and/or clarify a certain requirement [10, 13, 5, 14].

Issue-3: *Requirements prioritization*. To have a shortage of requirements is seldom a real issue, quite the opposite. This makes prioritization of the requirements a valuable tool in the initial stages of a project. There are several methods for prioritization [21, 22], one of the most well known is the Analytic Hierarchy Process or AHP for short [23]. The main issue with this method is scale-up issues, i.e. prioritizing of a lot of requirements is rather time consuming [21]. Getting the customers (or other departments) to give input, i.e. do some prioritization of their own, and show what is important to them, is an approach that could be beneficial, and combined with ones own prioritization efforts [24]. The exact method used depends on any number of factors, e.g. like amount of requirements, time devoted to prioritization and so on. The main issue here is not to recommend a prioritization method, but to recommend that a method be used.

Issue-4: *Requirements upkeep during & post project.* Requirements change. Making the requirements reflect these changes is crucial for several reasons. It is a prerequisite for being able to conduct tests based on the requirements (Issue-7). Furthermore in order to achieve any reuse (Issue-10) or re-prioritization of requirements this issue has to be resolved. Figure 5 illustrates an updated version (only triangulated issues) of dependencies between issues. Issue-4 is here dependent on Issue-6, i.e. that there is a clear and unambiguous definition of the roles and responsibilities pertaining to RE, this includes who should be responsible for making sure that the requirements are kept up to date.

Issue-6: *Roles and responsibilities RE process*. The importance of making roles and responsibilities clear is not an issue reserved for RE, but pertinent in any organization involved in software development [25]. One issue for RE is to identify the tasks to be conducted [5, 11], and then assign responsibility for them. Typically one would assume that there is one person responsible for the RE pertaining to a certain project, this does not however necessarily imply that this individual has to perform all tasks in question. The tasks could be delegated to any number of project team members. It is important that there be no question about who should perform what tasks. The delegation and division of tasks should be documented and rooted in the project organization in the same manner as the general project model.

Issue-7: *System tests performed against requirements.* Requirements are typically described as *what* is to be delivered to a customer [5]. By performing system tests on requirements it is possible to validate if the implemented features are really based on the requirements (functional requirements) [14], and if the system complies with other requirements (non-functional requirements) [14]. Making test scenarios/cases during initial requirements engineering can also be a good way of validating the requirements themselves at an early stage [5].

Issue-9: *RE process/methods*. All issues are included here. There are several RE processes suggested in literature [5, 11, 14], all more or less based on similar concepts and ideas. The main question for an organization to decide is what tasks are necessary to achieve their goals.

Issue-10: *Requirements reuse.* Reuse is a fairly common term in software engineering and often refers to the reuse of everything from code to architectures. The reuse of requirements however is not as commonly adopted [26]. This does not mean that the potential benefits of such a reuse are less, quite the opposite actually. By reusing requirements (traceability is a prerequisite, Issue-11, see Figure 5 [27]) everything from the requirement itself to analysis, design, implemented components, test cases, scenarios, and use cases etc. can be reused [5].

Issue-11: *Requirements traceability*. Several types of traceability could be identified in RE [11]. Only two are mentioned here, i.e. *Backward-from traceability* denoting a link from requirement to their source in other documents or people, and *Forward-from traceability* denoting a link from requirements to the design and indirectly to the implemented components [5, 11]. The first is important for verifying what sources there are for a certain requirement [28], and the second for making reuse of more that just the requirements themselves possible. There are however several views on requirements traceability [5, 11, 29] which should be studied before a distinction about what model for traceability should be implemented in an organization.

Issue-12: *Requirements version handling*. This issue could effectively be sorted under the umbrella of traceability [5, 11], but is separated due to the fact that this issue was identified separately from Issue-11 during the line interviews (see section 4.2.1.). Version handling of code in classes, components etc. is widely used in software engineering. The benefits are many, the most obvious one being the ability to go back and look at/use/compare to previous versions of an item. The same is true for a requirement, i.e. version handling enables an evolutionary view. In order to obtain this some kind of version handling system must be used, e.g. as a part of a requirements handling application.

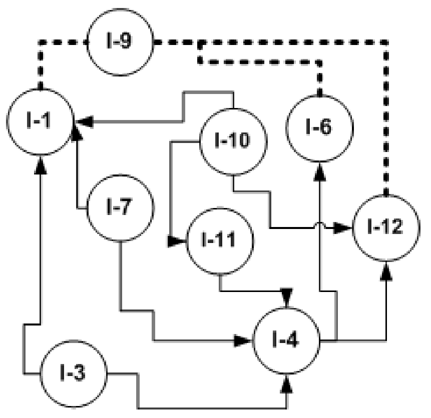

Figure 5. Dependency diagram (updated).

6 Conclusions

By conducting two separate studies, i.e. the project study and the line study, we were able to identify several improvement issues in DHR's RE process. Using the lightweight triangulation approach nine (out of a total of twelve) improvement issues were triangulated. The result was nine tangible issues that positively identified improvements to be made in the RE process at DHR.

A good enough RE process is crucial in order for development projects to be successful, it is however also crucial to have the ability to plan for the evolvement of the process. Process improvement is however often a costly enterprise tying up valuable resources during the improvement, and subsequent indoctrination of the improved process often means that substantial educational measures have to be taken. Ad hoc improvement measures, i.e. trying to improve the RE process as a whole, can mean that issues not crucial are improved/implemented, thus the process improvement costs are higher. Cost is a central concern for all organizations, and especially in the case of SME:s.

By identifying tangible and crucial improvement issues through eliciting information from personnel, management and documentation, and subsequently triangulating the issues identified, validate that the right problems are prioritized. A prioritization of issues gives the organization the raw material for the construction of a process improvement plan that addresses primarily crucial issues. Secondary issues, e.g. the ones identified during the studies but not triangulated, can be addressed at a later stage or the improvement approach could be used again.

It is also important to realize that the nature of the lightweight approach described in this paper bases its results on the knowledge and views of the people in the organization whose RE process is to be improved. This is beneficial in terms of rooting the coming changes in the organization at all levels.

Furthermore the usage of a lightweight triangulation approach to find improvement issues has a moderate price tag attached to it, as well as being an alternative to a more extensive evaluation process. This makes the use of a lightweight approach beneficial for SME:s. For a large enterprise the cost of using the lightweight approach would probably be higher. This is based on the assumption that the studies would be larger and the analysis more resource consuming, unless it is applied to a part of the enterprise. The investigation presented here took about 150 person-hours to complete.

The main risk in using the lightweight approach described in this paper is that issues may escape identification. This in turn can cause critical issues to remain unidentified and unattended. However, two things alleviate this risk. One, the identification of the dependencies between the issues was mapped, establishing an understanding of the interrelations of the issues in question. These dependencies help establish that there is not any crucial step missing as the issues are identified, as well as being a useful tool for the planning of the process improvement itself.

Second, following a process improvement plan, and the execution of such a plan, the studies and triangulation can be repeated. This would hopefully yield confirmation that the addressed issues were turned into non-issues, as well as identifying new improvement issues. The new issues could be issues not prioritized before, but issues missed the first time around could also be caught.

Future work consists of taking the next step, i.e. ascertaining what improvement issues are to be addressed first. This can be done using several different approaches. One is basically to base the priority of the issues on the relations in the dependency diagrams. Another is taking advantage of the expertise present in the organization that has been evaluated, i.e. making the decision up to the same people that were involved in the studies, and letting them prioritize the issues using some prioritization scheme, e.g. AHP.

Process improvement is an ongoing process with several steps, but the first one should be to identify what to improve.

7 Acknowledgements

The investigation described in this paper was conducted at DHR, in close cooperation with project and line personnel, as well as management. We would like to thank all involved for their help, patience, and enthusiasm for new ideas as well as seeing possibilities and not obstacles.

8 References

[1] I.K. Bray, *An Introduction to Requirements Engineering*, Addison-Wesley, Dorset, UK, 2002.

[2] R. Glass, *Software Runaways*, Prentice Hall, Harlow, UK, 1998.

[3] http://sme.cordis.lu/home/index.cfm, 2003-05-05.

[4] C. Robson, *Real World Research – Second edition*, Blackwell Publishers Ltd., Oxford, UK, 2002.

[5] I. Sommerville and P. Sawyer, *Requirements Engineering – A Good Practice Guide*, John Wiley & Sons, Chichester, UK, 2000.

[6] http://www.comp.lancs.ac.uk/computing/research/cseg/projects/reaims/index.html, 2003-05-05.

[7] The TickIT Guide – *Using ISO 9001:2000 for Software Quality Management System*, Construction, Certification and Continual Improvement, Issue 5.0, 2001.

[8] L. Bratthall and M. Jorgensen, "Can you Trust a Single Data Source Exploratory Software Engineering Case Study?", Kluwer, Netherlands, March 2002, pp. 9-26.

[9] B. Kitchenham and L. Pickard, "Case Studies for Method and Tool Evaluation", IEEE Software, IEE/IEEE, July 1995, pp. 52-62.

[10] I. Sommerville, *Software Engineering - 6th Edition*, Addison-Wesley, Essex, UK, 2001.

[11] T.B. Gorschek, K. Tejle, and M. Svahnberg, "A Study of the State of Requirements Engineering in Four Industry Cases", Proceedings of SERPS02, ISSN 1103-1581, Karlskrona, Sweden, 2002, pp. 111-119.

[12] C. Wohlin, P. Runeson, M. Höst, M.C. Ohlsson, B. Regnell and A. Wesslén, *Experimentation in Software Engineering – An Introduction*, Kluwer Academic Publishers, Boston, USA, 2000.

[13] G. Kotonya and I. Sommerville, *Requirements Engineering Processes and Techniques*, Wiley & Sons, Somerset, UK, 2001.

[14] S. Robertson and J. Robertson, *Mastering the Requirements Process*, Addison-Wesley, Harlow, UK, 1999.

[15] H. Sharp, A. Finkelstein, G. Galal, "Stakeholder Identification in the Requirements Engineering Process", Database and Expert Systems Applications, IEEE Computer Soc., 4/1999, pp. 387-391.

[16] K.S Hanks, J.C. Knight, and E.A. Strunk, "Erroneous Requirements: a Linguistic Basis for their Occurrence and an Approach to their Reduction", Proceedings. 26th Annual NASA Goddard Software Engineering Workshop 2001, IEEE Computer Soc., 2002, pp. 115-119.

[17] C. Rupp, "Requirements and Psychology", IEEE Software, IEE/IEEE, Issue 3, 2002, pp. 16-18.

[18] C. Potts, "Metaphors of Intent", Proceedings of Fifth IEEE International Symposium on Requirements Engineering, ISSN 1090705x, IEEE, 2001, pp. 31-38.

[19] I. Sommerville, T. Rodden, P. Sawyer, R. Bentley, and M. Twidale, "Integrating Ethnography Into the Requirements Engineering Process", Proceedings of IEEE International Symposium on Requirements Engineering 1993, IEEE Computer Soc., 1992, pp. 165-173.

[20] B. Potter, J. Sinclair, and D. Till, *An Introduction to Formal Specifications and Z – Second Edition*, Prentice Hall, Glasgow, UK, 1996.

[21] J. Karlsson, C. Wohlin, and B. Regnell, "An Evaluation of Methods for Prioritizing Software Requirements", Information and Software Technology, Issue 14-15, Elsevier, 1998, pp. 939-947.

[22] J. Karlsson, "Software Requirements Prioritizing", Proceedings of the Second International Conference on Requirements Engineering, ISBN 0818672528, IEEE Computer Soc. Press, 1996, pp. 110-116.

[23] T. Saaty and L. Vargas, *Models, Methods, Concepts & Applications of the Analytic Hierarchy Process*, Kluwer Academic Publishers, Boston, 2001.

[24] B. Regnell, M. Höst, J. Natt och Dag, P. Beremark, and T. Hjelm, " An Industrial Case Study on Distributed Prioritisation in Market-Driven Requirements Engineering for Packaged Software", Requirements Engineering, issue 1, Springer, 2001, pp. 51-62.

[25] K. Kivisto, "Roles of Developers as Part of a Software Process Model", Proceedings of the 32nd Annual Hawaii International Conference on Systems Sciences, IEEE Computer. Soc., 1999, pp. 1-19.

[26] W. Lam, "A Case-study of Requirements Reuse through Product Families", Annals of Software Engineering, issue 5, 1998, Kluwer, 1998, pp. 253-277.

[27] O. Roudies and M. Fredj, "A Reuse Based Approach for Requirements Engineering", ACS/IEEE International Conference on Computer Systems and Applications, IEEE Computer Soc., 2001, pp. 448-450.

[28] O. Gotel and A. Finkelstein, "Extended Requirements Traceability: Results of an Industrial Case Study", Proceedings of the Third IEEE International Symposium on Requirements Engineering, IEEE Computer Soc., 1997, pp. 169-178.

[29] B. Ramesh and M. Jarke, "Toward Reference Models for Requirements Traceability", IEEE Transactions on Software Engineering, issue 1, IEE/IEEE, 2001, pp. 58-93.

[30] J. Calvo-Manzano Villalón, et al., “Experiences in the Application of Software Process Improvement in SMES”, Software Quality Journal, issue 3, Kluwer, 2002, pp. 261-273.

9 Author CVs

Tony Gorschek,

T. Gorschek is a PhD student in software engineering at the Department of Software Engineering and Computer Science at Blekinge Institute of Technology in Sweden. Prior to this, he has worked as CTO (chief technical officer) for a system development company, and later as a consultant in the area of systems engineering and adaptation. He has a M.Sc. in Computer Science from Blekinge Institute of Technology. His research interests mainly include software process improvement and requirements engineering.

Dr. Claes Wohlin

Dr. Wohlin is a professor of software engineering at the Department of Software Engineering and Computer Science at Blekinge Institute of Technology in Sweden. Prior to this, he has held professor chairs in software engineering at Lund University and Linköping University. He has a Ph.D. in Communication Systems from Lund University. His research interests include empirical methods in software engineering, software metrics, software quality and systematic improvement in software engineering. Claes Wohlin is the principal author of the book "Experimentation in Software Engineering – An Introduction" published by Kluwer Academic Publishers in 2000. He is co-editor-in-chief of the journal of Information and Software Technology published by Elsevier. Dr. Wohlin is on the editorial boards of Empirical Software Engineering: An International Journal and Software Quality Journal. He is a member of IEEE and ACM.