# Using Rational Unified Process in an SME – A Case Study

**Conference Paper** *in* Lecture Notes in Computer Science · November 2005

DOI: 10.1007/11586012_14 · Source: DBLP

Geir Kjetil Hanssen
SINTEF
**98** PUBLICATIONS **1,910** CITATIONS
SEE PROFILE

Hans Westerheim
SINTEF
**22** PUBLICATIONS **150** CITATIONS
SEE PROFILE

Finn Olav Bjørnson
SINTEF
**38** PUBLICATIONS **971** CITATIONS
SEE PROFILE

# Using Rational Unified Process in an SME – A Case Study

Geir Kjetil Hanssen[1], Hans Westerheim[1], Finn Olav Bjørnson[2]

[1]SINTEF ICT, N-7465 Trondheim, Norway
{geir.kjetil.hanssen, hans.westerheim}@sintef.no
[2]Norwegian University of Science and Technology, N-7491 Trondheim, Norway
bjornson@idi.ntnu.no

**Abstract.** The Rational Unified Process (RUP) is a comprehensive software development process framework emphasizing use-cases, architecture focus and an iterative approach. RUP is widely known and many organizations have tried to adopt it. Being a framework, RUP has to, in some way, be tailored to the specific context of use, no software development project is alike. This paper presents a case study of a Norwegian SME that tried to adopt RUP in the simplest way, by introducing the methodology by providing comprehensive documentation and some simple training. Our study shows that the use of RUP had some positive effects but also that the use has been scattered. Interviews with users of RUP show that there is a great need of better training and practical support in getting most value out of RUP. The key message is that if you consider taking RUP into use you have to invest resources in it. Training and support are key success factors.

## 1. Introduction

The Rational Unified Process (RUP) is a software development process framework consisting of a more or less complete set of process elements for software development projects [1]. RUP defines a software development project as a set of disciplines, e.g. requirements handling, implementation etc., running from start to end through the whole project life cycle divided in a set of project phases. A project is performed by a group of actors, each having one or more well defined roles. Each role participates in one or more activities producing one or more artifacts. A discipline can run in iterations, that is, repetitions within a phase. Activities, roles and artifacts are the basic process elements of RUP. RUP is a prescriptive and plan driven methodology. As RUP is a comprehensive framework covering most aspects of a software development process it means that it in some way must be adapted to the situation of use, either ad-hoc for each project or in advance to produce a company wide standard.

In this paper we present a case study that describes the use of RUP in a company where no restrictions or guidelines were put on the use of RUP. The project managers and senior developers were given courses in RUP, and RUP Online (an electronic process guide on web) was purchased and installed. No common guidance for the use of RUP in projects was given. The company had no defined goals for introducing RUP; it was basically based on a belief that RUP would increase the professionalism in the company. The study has been conducted within a smaller Norwegian software development company. Three researchers have followed the company during a period of three years. This paper describes the experience from using RUP and derives some key conclusions that may be of use for others considering the use of RUP.

The paper has the following structure:

- The research method is described (collection of empirical data and data analysis).
- The research context of the case study, that is, the company, is described.
- The results part documents information and data collected. This includes descriptions of the usage of RUP as well as elements laying the ground for the forthcoming tailoring of RUP. Further on the results from the analysis of four projects and five interviews are documented.
- A discussion trying to clarify the key points from the analysis and giving a conclusion.

## 2. Research Method

### 2.1 Data collection

The study has taken the form of a case study [2]. The research has been conducted by three external researchers mainly using project managers and software developers from the company as a source for data.

**The first set of data** was collected by interviewing project managers representing four projects. Prior to this series of interviews, the researchers prepared a spreadsheet that had a row for each role, activity and artifact described by RUP, grouped by the disciplines defined by RUP. A column was allocated to each project. The researcher conducting the interview asked the project manager about the use of every single role, activity and artifact in the actual project. If the element was used in the project as described by RUP, the actual cell was colored. If the item was used as described by RUP, but changed or replaced by a tailored element, the cell was colored and a comment was written about the change from the original item description in RUP. If the element was not used at all in the project, the cell was left blank.

**The second set of data** was collected by the means of semi-structured interviews with five other employees (each having experience with RUP from several various projects). The respondents had the following main responsibilities: 1) developer, 2) developer/project manager, 3) developer/project manager/test manager, 4) project manager/requirements engineer and 5) customer contact. Prior to the interviews, the researchers developed an interview guide. The guide consisted of questions with focus on their personal experience with using RUP across multiple projects to document a broader experience. The guide was open, allowing the respondents to freely discuss their experience.

These interviews were recorded and transcribed by the researchers. The transcriptions were reviewed by the interviewed objects, and possible corrections and clarifications were made.

### 2.2 Data Analysis

The spreadsheet documenting the use of RUP and the transcribed interviews were basis for the data analysis. From the beginning it was clear that the researchers were to use qualitative data analysis methods due to the nature of the data collected [3, 4].

**Analysis of the spreadsheet.** The spreadsheet was printed in 25% size of its actual size. This was done to get an *overview* of the RUP usage. The overview gave a clear visual picture of what parts of RUP which were really used. The comments in the spreadsheet were read through. The researchers tried to match comments with the non-use, to see if there might be some statements supporting the lack of usage. The RUP usage for the projects were also compared to the project definition, the scope of the project and the type of customer as a starting point for an understanding of the actual use and non-use of RUP elements.

**Analysis of the interviews.** The researchers used the constant comparison method [3] to identify the factors affecting the use of RUP among the project managers and senior developers. All the transcriptions were printed out, and each of the researchers got one copy each. The single transcriptions were read individually, and the researcher tagged statements in the documents which said something about use or non-use of RUP, reasons for using or not using RUP, and also positive and negative aspects with RUP itself. Then the researchers had a common work shop where the individual tagging were put onto a white board, and then compared. The comparison was the basis for a common summary of the interviews.

The main motivation for selecting this approach to data collection and data analysis was that the researchers did not have any pre-information about the use of RUP, and therefore no assumptions or hypothesis to test, thus a qualitative approach seemed appropriate. This motivation was supported by the relative low number of data points available from such a small company.

## 3. Research context

The company described in this case is today a Norwegian software consultancy company with 50 employees, located in two different geographic offices.

They are mainly developing software systems with heavy back-end logic and often with a web front-end, typically portals. However, they also develop lighter solutions with most emphasis on the front-end.

The company acts as an independent software supplier, though there are close relationships to the biggest customers. Of the 50 employees today, 35 are working as software developers. Java and J2EE are used as the main development platform. The domain of which the company develops software is mainly for the banking and finance sector, as well as for public sector. The company has run 50 development projects within the bank and finance sector the last twelve years, and about 30-40 projects within the public sector the last 15 years.

Four employees are certified RUP-mentors acting as advisors in other SW-organizations, in addition to this they also used to run training courses in RUP as part of their partnership with Rational (now IBM Rational).

During the work described in this paper the company was declared bankrupt, and then restarted with new owners, but with the same employees. The data collection (using the spreadsheet) took place before the bankruptcy; the interviews took place about six months after the company was restarted.

## 4. Results

### 4.1 Interview round 1: Documenting the use of RUP

The four projects investigated had a scattered use of RUP. Interviewing the project leaders we documented the projects per phase to see which process elements were used and which were not and the corresponding reasons for that. In the following we present a summary per phase for each project (named project A to D).

The business modeling discipline

Project A was about porting functionality, no new functionality was introduced, thus not needing business modeling. For project B the customer had provided a business use case that was sufficient. Project C was developing software to be integrated with other systems. The business modeling discipline was used to clarify these interfaces. Project D had a business modeling discipline although it was not performed exactly as described by RUP.

The requirements discipline

Project A used the discipline partly to specify requirements for how to join the user interface of several systems. The other three projects used the requirements discipline quite extensively.

The analysis and design discipline

The elements in the discipline were partly used for all four projects. However there was a lot of adoption to the project context.

### The implementation discipline

The use of the process elements in the implementation discipline was scattered. Although all four projects used it, project A used it briefly, project C and D used it extensively.

### The test discipline

Project B and D had an extensive use of the process elements in the test discipline while the two other projects did not follow RUP for testing.

### The deployment discipline

Project A had deployment activities but these were not done according to RUP. Project B had at the time of the interview not got to this. The deployment in project C was done partly by the customer that had responsibility for most of the activities. Project D did utilize most elements from RUP.

### The configuration and change management discipline

Project A did not follow RUP at all; this was however done using a specialized system guiding the process of configuration and change management. Project B and C did use RUP pretty extensively for this discipline. For project D the customer handled this responsibility following other procedures than described by RUP.

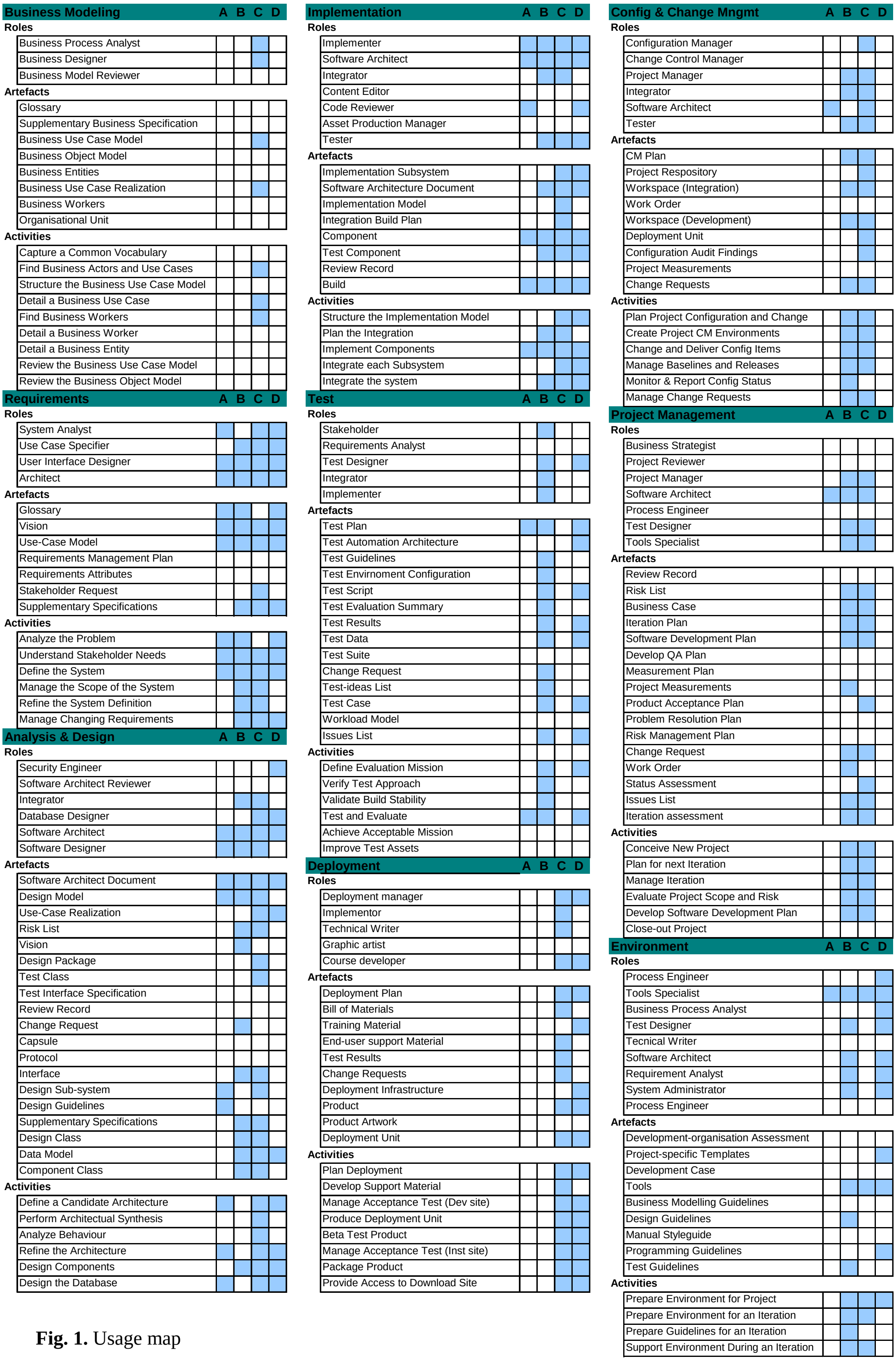
### The project management discipline

Project A did not follow RUP except for the use of the software architect role. Project B and C did use most of the process elements from RUP. In the case of project D the customer had the project management responsibility themselves.

### The environment discipline

Project A had merely no use of this discipline. Project B used most process elements. Project C used only a few but project D used several.

By mapping the use of process elements for the four projects we made a visual map documenting the use of each process element in RUP, ordered by the eight disciplines that RUP describes (see fig. 1).



| Business Modeling | A | B | C | D |
|---|---|---|---|---|
| **Roles** | | | | |
| Business Process Analyst | | | ■ | |
| Business Designer | | | ■ | |
| Business Model Reviewer | | | | |
| **Artefacts** | | | | |
| Glossary | | | | |
| Supplementary Business Specification | | | | |
| Business Use Case Model | | | ■ | |
| Business Object Model | | | | |
| Business Entities | | | | |
| Business Use Case Realization | | | ■ | |
| Business Workers | | | | |
| Organisational Unit | | | | |
| **Activities** | | | | |
| Capture a Common Vocabulary | | | | |
| Find Business Actors and Use Cases | | | ■ | |
| Structure the Business Use Case Model | | | | |
| Detail a Business Use Case | | | ■ | |
| Find Business Workers | | | ■ | |
| Detail a Business Worker | | | | |
| Detail a Business Entity | | | | |
| Review the Business Use Case Model | | | | |
| Review the Business Object Model | | | | |

| Requirements | A | B | C | D |
|---|---|---|---|---|
| **Roles** | | | | |
| System Analyst | ■ | | ■ | ■ |
| Use Case Specifier | | ■ | ■ | ■ |
| User Interface Designer | ■ | ■ | ■ | ■ |
| Architect | ■ | ■ | ■ | ■ |
| **Artefacts** | | | | |
| Glossary | ■ | ■ | | ■ |
| Vision | ■ | ■ | ■ | ■ |
| Use-Case Model | ■ | ■ | ■ | ■ |
| Requirements Management Plan | | | | |
| Requirements Attributes | | | | |
| Stakeholder Request | | | ■ | |
| Supplementary Specifications | | ■ | ■ | ■ |
| **Activities** | | | | |
| Analyze the Problem | ■ | ■ | | ■ |
| Understand Stakeholder Needs | ■ | ■ | ■ | ■ |
| Define the System | ■ | ■ | ■ | ■ |
| Manage the Scope of the System | | ■ | ■ | |
| Refine the System Definition | | ■ | ■ | |
| Manage Changing Requirements | | ■ | ■ | ■ |

| Analysis & Design | A | B | C | D |
|---|---|---|---|---|
| **Roles** | | | | |
| Security Engineer | | | | ■ |
| Software Architect Reviewer | | | | |
| Integrator | | ■ | ■ | |
| Database Designer | | | ■ | ■ |
| Software Architect | ■ | ■ | ■ | ■ |
| Software Designer | ■ | ■ | ■ | |
| **Artefacts** | | | | |
| Software Architect Document | ■ | ■ | ■ | ■ |
| Design Model | ■ | ■ | ■ | |
| Use-Case Realization | | | ■ | ■ |
| Risk List | | ■ | ■ | |
| Vision | | ■ | | |
| Design Package | | | ■ | |
| Test Class | | | ■ | |
| Test Interface Specification | | | | |
| Review Record | | | | |
| Change Request | | ■ | | |
| Capsule | | | | |
| Protocol | | | | |
| Interface | | ■ | ■ | |
| Design Sub-system | ■ | | ■ | |
| Design Guidelines | ■ | | | |
| Supplementary Specifications | | ■ | ■ | |
| Design Class | | ■ | ■ | |
| Data Model | | ■ | ■ | ■ |
| Component Class | | ■ | ■ | |
| **Activities** | | | | |
| Define a Candidate Architecture | ■ | | ■ | ■ |
| Perform Architectual Synthesis | | | ■ | |
| Analyze Behaviour | | | ■ | |
| Refine the Architecture | ■ | | ■ | ■ |
| Design Components | | ■ | ■ | ■ |
| Design the Database | ■ | | ■ | ■ |

| Implementation | A | B | C | D |
|---|---|---|---|---|
| **Roles** | | | | |
| Implementer | ■ | ■ | ■ | ■ |
| Software Architect | ■ | ■ | ■ | ■ |
| Integrator | | ■ | ■ | |
| Content Editor | | | | |
| Code Reviewer | ■ | | | ■ |
| Asset Production Manager | | | | |
| Tester | | ■ | ■ | ■ |
| **Artefacts** | | | | |
| Implementation Subsystem | | | ■ | ■ |
| Software Architecture Document | | ■ | ■ | ■ |
| Implementation Model | | | ■ | |
| Integration Build Plan | | | ■ | |
| Component | ■ | ■ | ■ | ■ |
| Test Component | | ■ | ■ | ■ |
| Review Record | | | | |
| Build | ■ | ■ | ■ | ■ |
| **Activities** | | | | |
| Structure the Implementation Model | | | ■ | ■ |
| Plan the Integration | | ■ | ■ | |
| Implement Components | ■ | ■ | ■ | ■ |
| Integrate each Subsystem | | | ■ | ■ |
| Integrate the system | | ■ | ■ | ■ |

| Test | A | B | C | D |
|---|---|---|---|---|
| **Roles** | | | | |
| Stakeholder | | ■ | | |
| Requirements Analyst | | | | |
| Test Designer | | ■ | | ■ |
| Integrator | | ■ | | |
| Implementer | | ■ | | |
| **Artefacts** | | | | |
| Test Plan | ■ | ■ | | ■ |
| Test Automation Architecture | | | | ■ |
| Test Guidelines | | ■ | | |
| Test Envirnoment Configuration | | ■ | | |
| Test Script | | ■ | | ■ |
| Test Evaluation Summary | | ■ | | |
| Test Results | | ■ | | ■ |
| Test Data | | ■ | | ■ |
| Test Suite | | | | |
| Change Request | | ■ | | |
| Test-ideas List | | ■ | | |
| Test Case | | ■ | | ■ |
| Workload Model | | | | |
| Issues List | | ■ | | ■ |
| **Activities** | | | | |
| Define Evaluation Mission | | ■ | | ■ |
| Verify Test Approach | | ■ | | |
| Validate Build Stability | | ■ | | |
| Test and Evaluate | ■ | ■ | | ■ |
| Achieve Acceptable Mission | | | | |
| Improve Test Assets | | | | |

| Deployment | A | B | C | D |
|---|---|---|---|---|
| **Roles** | | | | |
| Deployment manager | | | ■ | ■ |
| Implementor | | | ■ | |
| Technical Writer | | | ■ | |
| Graphic artist | | | | |
| Course developer | | | ■ | ■ |
| **Artefacts** | | | | |
| Deployment Plan | | | ■ | ■ |
| Bill of Materials | | | ■ | |
| Training Material | | | | ■ |
| End-user support Material | | | ■ | |
| Test Results | | | ■ | |
| Change Requests | | | ■ | |
| Deployment Infrastructure | | | | ■ |
| Product | | | ■ | ■ |
| Product Artwork | | | | |
| Deployment Unit | | | ■ | ■ |
| **Activities** | | | | |
| Plan Deployment | | | ■ | ■ |
| Develop Support Material | | | ■ | |
| Manage Acceptance Test (Dev site) | | | ■ | ■ |
| Produce Deployment Unit | | | ■ | ■ |
| Beta Test Product | | | ■ | ■ |
| Manage Acceptance Test (Inst site) | | | ■ | ■ |
| Package Product | | | ■ | ■ |
| Provide Access to Download Site | | | ■ | ■ |

| Config & Change Mngmt | A | B | C | D |
|---|---|---|---|---|
| **Roles** | | | | |
| Configuration Manager | | | ■ | |
| Change Control Manager | | | | |
| Project Manager | | ■ | ■ | |
| Integrator | | ■ | ■ | |
| Software Architect | ■ | | ■ | |
| Tester | | ■ | ■ | |
| **Artefacts** | | | | |
| CM Plan | | ■ | ■ | |
| Project Respository | | | ■ | |
| Workspace (Integration) | | ■ | ■ | |
| Work Order | | | | |
| Workspace (Development) | | ■ | ■ | |
| Deployment Unit | | | ■ | |
| Configuration Audit Findings | | | ■ | |
| Project Measurements | | | | |
| Change Requests | | ■ | ■ | |
| **Activities** | | | | |
| Plan Project Configuration and Change | | ■ | ■ | |
| Create Project CM Environments | | ■ | ■ | |
| Change and Deliver Config Items | | ■ | ■ | |
| Manage Baselines and Releases | | ■ | ■ | |
| Monitor & Report Config Status | | ■ | | |
| Manage Change Requests | | ■ | ■ | |

| Project Management | A | B | C | D |
|---|---|---|---|---|
| **Roles** | | | | |
| Business Strategist | | | | |
| Project Reviewer | | | | |
| Project Manager | | ■ | ■ | |
| Software Architect | ■ | ■ | ■ | |
| Process Engineer | | | | |
| Test Designer | | ■ | ■ | |
| Tools Specialist | | ■ | ■ | |
| **Artefacts** | | | | |
| Review Record | | | | |
| Risk List | | ■ | ■ | |
| Business Case | | ■ | ■ | |
| Iteration Plan | | ■ | ■ | |
| Software Development Plan | | ■ | ■ | |
| Develop QA Plan | | | | |
| Measurement Plan | | | | |
| Project Measurements | | ■ | | |
| Product Acceptance Plan | | | ■ | |
| Problem Resolution Plan | | | | |
| Risk Management Plan | | | | |
| Change Request | | ■ | ■ | |
| Work Order | | ■ | | |
| Status Assessment | | | ■ | |
| Issues List | | ■ | ■ | |
| Iteration assessment | | ■ | ■ | |
| **Activities** | | | | |
| Conceive New Project | | ■ | ■ | |
| Plan for next Iteration | | ■ | ■ | |
| Manage Iteration | | ■ | ■ | |
| Evaluate Project Scope and Risk | | ■ | ■ | |
| Develop Software Development Plan | | ■ | ■ | |
| Close-out Project | | | | |

| Environment | A | B | C | D |
|---|---|---|---|---|
| **Roles** | | | | |
| Process Engineer | | | | ■ |
| Tools Specialist | ■ | ■ | ■ | ■ |
| Business Process Analyst | | | | ■ |
| Test Designer | | ■ | | ■ |
| Tecnical Writer | | | | |
| Software Architect | | ■ | | ■ |
| Requirement Analyst | | ■ | | ■ |
| System Administrator | | ■ | | ■ |
| Process Engineer | | | | |
| **Artefacts** | | | | |
| Development-organisation Assessment | | | | |
| Project-specific Templates | | | | ■ |
| Development Case | | | | |
| Tools | | ■ | ■ | ■ |
| Business Modelling Guidelines | | | | |
| Design Guidelines | | ■ | | |
| Manual Styleguide | | | | |
| Programming Guidelines | | | | ■ |
| Test Guidelines | | ■ | | |
| **Activities** | | | | |
| Prepare Environment for Project | | ■ | ■ | ■ |
| Prepare Environment for an Iteration | | ■ | ■ | |
| Prepare Guidelines for an Iteration | | ■ | | |
| Support Environment During an Iteration | | ■ | ■ | |

**Fig. 1.** Usage map

### 4.2 Interview round 2: Experiences with using RUP

Five project participants with experience from several projects were interviewed to document positive and negative experiences from their use of RUP as well as any improvement suggestions. Note that these interviews are not related to the interviews in round one.

Some of the respondents had experience with more than one role and project type. The five persons interviewed had the following background:

- Respondent 1: Roles: Developer, project manager and test manager. Project types: Web applications with backend logic.
- Respondent 2: Roles: Developer, project manager and test manager (often combined). Project types: Web applications.
- Respondent 3: Roles: Project manager, requirements manager. Project types: Publication systems, banking systems.
- Respondent 4: Roles: Developer. Project types: Mostly system maintenance.
- Respondent 5: Roles: Key account manager. Project types: Secure systems.

Of the five respondents three defined their RUP knowledge as 'good', one as 'medium' and one as 'little'. Following is a summary of common statements from the interview transcriptions showing which respondents having which statements.

| | | Interview respondents | | | | |
|---|---|---|---|---|---|---|
| +/- | Nodes from interview coding | 1 | 2 | 3 | 4 | 5 |
| Positive experience | The RUP training was good | | | | x | |
| | Roles defined by RUP | x | x | | | |
| | Important to have a supporting process | | | | x | |
| | Used inception and elaboration [with success] | | | | x | |
| | SW maintenance projects uses RUPs guidelines for transition between phases/milestones | x | | | | |
| | Templates and role definitions are good checklists | x | | | | |
| | Want to be better at using RUP | x | x | x | x | x |
| | Reasonable division in phases and iterations | x | | | | |
| Negative experience | To extensive for small projects | x | x | x | x | |
| | Too document-driven | | | | x | |
| | Too much focus on just the development | | | | | x |
| | Missing roles for customer contact prior to and past the development project | | | | | x |
| | Requires good knowledge [of RUP] | | | | | x |
| | Missing a common standard of use | | x | x | | |
| | Does not fit a software maintenance processes | x | | | | |
| | Miss adaptation to extreme programming | x | | | | |
| | We do not evaluate our use of RUP | x | | | | |
| | Continues with old practice | | | | x | |
| | We have not changed our practice after RUP was introduced | | x | | | |
| | I have no progress [as a software professional] | x | | x | | |
| | Missing follow-up during projects | | x | x | | |
| | Hard to understand RUP | | x | | | |

**Table 1.** Interviews summary

Note that this list is a collection of all statements found relevant to the use of RUP. Some are clear and can be generalized; others are specific to a single project. The definition of the nodes is based on an interpretation of the interviews (due to the constant comparison method).

Besides this overview of experience using RUP, the respondents also had improvement suggestions:

- RUP should be used in a regular manner through the whole project (avoiding deep focus in only parts of the project)
- Projects must be guided in the use of RUP
- Web-projects need more specialized support than RUP can offer
- Establish a project manager forum (for learning and experience exchange)
- Avoid the use of RUP in the case of software maintenance
- Offer support in using and adapting RUP

## 6. Discussion and conclusion

Offering RUP out-of-the-box leaves all the responsibility of tuning RUP to each individual project. This may cost both time and resources. Good knowledge on RUP is also needed. As the results from interview round one show, the use of RUP is scattered and deviates partly from the RUP guidelines. Project participants seems to end up using some RUP elements mixed with old practice, not as a consequence of deliberate decisions but as a consequence of low knowledge of RUP and how to adapt it.

The phases (and disciplines) of RUP covers the complete lifecycle of a software development project. However, in a real context, as the interviews show, the customer often has done some part of the job initially following an internal process. This may affect the use of RUP later on in the project.

Looking at the results from interview round two we see that most respondents support the idea (in general) of having a guiding process that includes role descriptions and regulates the work in disciplines, phases and iterations. However, all of the respondents feel that they need to and want to be better at using RUP. The reason for this may be that RUP is extremely comprehensive and that the task of fitting this framework to a project may be overwhelming. We also see that four of the respondents find RUP too comprehensive for small projects. This indicates a definitively need for tailoring of RUP in advance of use in projects. Two respondents also miss a common practice for the use of RUP, also indicating the need of a general tailoring of RUP.

In general, the interview results show that providing RUP just in the form of the full documentation (in this case RUP Online – right out of the box) have negative effects, at least not as good effects as one would believe in advance. It is perceived as too comprehensive and the users have problems finding the parts that would benefit their project. The consequence may be avoidance of use or even worse, wrong use. Two respondents claim that they have not changed their practice of developing software after RUP became available. This resembles with known acceptance models[5]; the methodology must be perceived as useful (will using RUP enhance the job performance?) and it must be perceived as easy to use (will using RUP require low effort?).

Besides doing a thorough adaptation in advance to increase usefulness and ease of use, projects also need practical guidance throughout the project; two respondents miss this type of support, this is also on the list of improvement suggestions. Introducing guidance and mentoring would both improve the degree of use and the effect of use of RUP as well as it would serve as a experience transfer mechanism.

**Conclusion:** The basic learning from this case study is that a methodology or framework (such as RUP) can not be provided “as is” without experiencing low/wrong use. The users of the methodology need to keep their focus on doing their job (developing software), not struggling to understand the theory. This is actually what the RUP documentation says, but that many unfortunately forget. Introducing a methodology such as RUP is an investment beyond the license fee. In this case the outcome could have been better if the introduction of RUP was carefully managed and not left as an autonomous effort in each project.

**A comment:** The learning from this study made the company decide to initiate a RUP adaptation process to provide their employees with better process support. This work is described in [6].

## 7. Further Research

The research reported in this paper, and also in other papers has put emphasis on the challenges in implementing and tailoring RUP for use in an organization [6-8]. Implementing a process framework like RUP can be looked upon as implementation of a new technology in an organization. It would therefore be of interest to study such implementations in spite of technology acceptance models [5] and investigate the success factors of tailoring and introduction of methodologies.

## Acknowledgements

The authors would like to thank the participants from the case company. We would also like to thank the SPIKE-project (funded by the Research Council of Norway) for funding and support.

## References

1. Krutchen, P., *The Rational Unified Process: An Introduction*. 2nd ed. 2000: Addison-Wesley. 298.
2. Yin, R.K., *Case Study Research - Design and Methods*. Applied Social Research Methods Series, ed. D.S. Foster. 1994: SAGE Publications.
3. Seaman, C., *Qualitative methods in empirical studies in software engineering.* IEEE Transactions on Software Engineering, 1999. **25**(4): p. 557-572.
4. Avison, D., et al., *Action Research.* Communications of the ACM, 1999. **42**(1): p. 94-97.
5. Riemenschneider, C.K.H., et al., *Explaining software developer acceptance of methodologies: a comparison of five theoretical models.* Software Engineering, IEEE Transactions on, 2002. **28**(12): p. 1135-1145.
6. Westerheim, H. and Hanssen, G.K., *The Introduction and Use of a Tailored Unified - A Case Study*. in *31st EUROMICRO CONFERENCE on Software Engineering and Advanced Applications (SEAA)*. 2005. Porto, Portugal: IEEE.
7. Bergström, S., Råberg, L., *Adopting the Rational Unified Process.* 2004, Addison-Wesley. p. 165-182.
8. Hanssen, G.K., et al., *Tailoring RUP to a defined project type: A case study*. in *6th International Conference on Product Focused Software Process Improvement, PROFES.* 2005. Oulo, Finland: Springer.