The implementation, adaptation, and use of the Rational Unified Process at Volvo Information Technology - a case study

(HS-IDA-EA-02-203)

Guðmundur Hallgrímsson
(a99gudha@student.his.se)

Department of Computer Science
University of Skövde, Box 408
S-54128 Skövde, SWEDEN

Final Year Project in Software Engineering, Spring 2002.

Supervisor: Per Backlund

Supervisor at Volvo IT: Ann-Britt Blom Karlsson

Examiner: Björn Lundell

The implementation, adaptation, and use of the Rational Unified Process at Volvo Information Technology - a case study

Submitted by Guðmundur Hallgrímsson to University of Skövde as a dissertation for the degree of B.Sc., in the Department of Computer Science.

2002-06-07

I certify that all material in this dissertation that is not my own work has been identified and that no material is included for which a degree has previously been conferred on me.

Signed: _____

The implementation, adaptation, and use of the Rational Unified Process at Volvo Information Technology - a case study

Guðmundur Hallgrímsson (a99gudha@student.his.se)

Abstract

The use of systems development methods are, by many, seen as the way to solve development problems, decrease development time, and improve the quality of software systems. Despite this, little is known about how development methods are actually used in the software industry. The aim of this project is to investigate how a widespread development method is implemented and used in an organisational setting.

The result of this project is a case study description of how Volvo Information Technology implements, adapts, and uses the commercial development method Rational Unified Process® (RUP®) in combination with other methods. The implementation is centrally administered and done incrementally over several years in order to build competence in the organisation. RUP is also adapted to the specific situation of the organisation, each division, each development project, and even adapted by individual developers.

Keywords: Software engineering, information systems, systems development, methods, methodologies, situational methods, method fragments, case study.

Table of Contents

List of Figures

List of Tables

1 Introduction

Software engineering is concerned with software systems built by teams rather than by individuals, uses engineering principles in the development of these systems and includes both technical and non-technical aspects (Sommerville, 1992, p. 2). Software systems are being developed in an increasingly complex business and technological environment (Wynekoop & Russo, 1997, p. 47). The methods and techniques used to develop software keep evolving but the software-related problems also intensify (Pressman, 1997, p. 7). Many researchers see the solution to the software problems in terms of increased control and more widespread adoption of development methods (Fitzgerald, 1996).

A "method is an approach to perform a systems development project, based on a specific way of thinking, consisting of directions and rules, structured in a systematic way in development activities with corresponding development products" (Brinkkemper, 1996, p. 275). Methods are therefore supposed to solve problems associated with ad hoc development, by guiding and structuring the development process and thereby facilitating project management.

It has been estimated that there are more than a thousand published development methods (Jayaratna, 1994, in Fitzgerald, 1996). Many of these methods are created by academics and published in the form of articles (e.g. Multiview2 (Avison et al., 1998)) or books (e.g. OMT (Rumbaugh et al., 1991)). Others can be documented in-house methods in organisations (e.g. Cork Organisational Standard Software Process (OSSP) (Fitzgerald et al., 2001). Such methods are often only used by the organisation itself, but can be used in cooperation between different organisations. There also exist methods that have become or been created as commercial products. An example of this is the Rational Unified Process®[1] (RUP®) (e.g. Kruchten, 1999).

Studies have shown that only 6% of those using methods use them rigorously as they are prescribed (Fitzgerald, 1996; Russo et al., 1996). This indicates that majority of method users tailor the methods to meet the specific needs of the organisation and the situation of each project. Other research has supported this and indicated that there is a wide difference between the formalised sequence of steps and stages prescribed by a methodology, and the method-in-action uniquely enacted for each development project (Fitzgerald, 1997). Even when developers are supposed to use a particular method, they use the method when it suits the situation and depart from it when the situation demands a different approach (Power & Richardson, 1996, p. 250).

Many researchers have criticised the lack of empirical research of the use of methods (e.g. Baskerville & Stage, 2001; Fitzgerald, 1994, 1996; Wynekoop & Russo, 1997). This implies that more empirical research is needed to understand the processes underlying development method usage in today's environment, which can in turn form the basis for the development and/or adaptation of methods in the future.

This project will therefore investigate the use of a development method in an organisational context. The project is undertaken in the hope of adding valuable knowledge to this field of research by increasing the understanding of how practitioners actually use methods. To be more precise, this project is focused on a

[1] Rational, Rational Unified Process, and RUP are trademarks or registered trademarks of Rational Software Corporation in the United States and/or other countries.

case study description of how the large organisation Volvo Information Technology implements, adapts, and uses the commercial development method Rational Unified Process in combination with other methods.

The rest of this report is organized as follows. First, software engineering, software problems, development approaches, and research on development methods will be discussed in a background chapter (chapter 2). Then the problem to be investigated is described, and project's aim and objectives discussed (chapter 3). From this follows a discussion on the approach and techniques used to investigate the stated problem (chapter 4). Chapter 5 describes the Rational Unified Process, in order to make the discussions in following chapters make sense to those not previously familiar with the method. Chapter 6 presents the organisational setting and the results of the case study. Chapter 7 presents an analysis of the results. The results are put into context of other research and the recommendations of the method vendor. Chapter 8 summarizes and discusses the results and points to possible future work on the research topic. At the end of the report there is a reference list that identifies all work that has been cited in the text. Appendices are discussed and explained at the appropriate locations in the report.

2 Background

The purpose of this section is to give a detailed presentation of the field of software engineering, its problems, and the use of development methods. First, software engineering and problems associated with software engineering will be described. Approaches for systems development and especially development methods, as proposed solutions to problems, will then be described. Existing research on methods will be introduced and discussed. Finally, the need for further research will be discussed.

2.1 Software engineering

There exist many possible definitions of the term software engineering (SE) but the most common factors are that software engineering is concerned with software systems built by teams rather than by individuals, uses engineering principles in the development of these systems, and includes both technical and non-technical aspects (Sommerville, 1992, p. 2). Software engineers must therefore have the skills not only to write code, but also to communicate orally and in writing, and be able to work efficiently in teams.

Software is not just the programs themselves but also the documentation necessary to develop, install, use, and maintain those programs. The software is usually only a part of a system to manipulate information or control some surroundings (e.g. a manufacturing plant). The scope of this dissertation is not meant to be limited to so-called information systems (IS), but rather to include any type of software systems.

Pressman (1997, p. 23) describes software engineering as a layered technology (Figure 1). The supporting bedrock is an organisational commitment to a quality focus. The foundation for software engineering is a process layer, and the software engineering process is the glue that holds the technology layers together and enables rational and timely development of computer software.

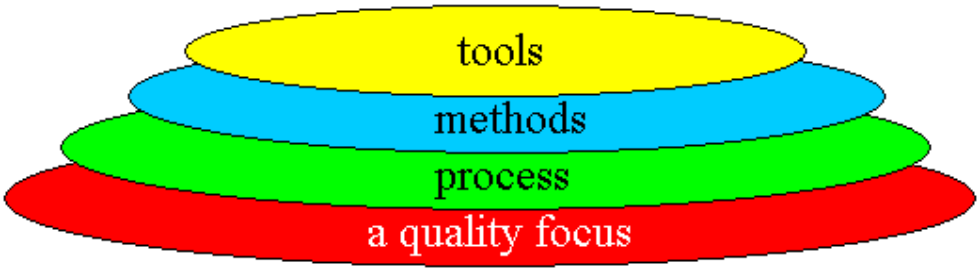

Figure 1 Software engineering layers (adapted from Pressman, 1997, p. 23)

The process defines a framework for a set of key process areas (e.g. project planning, project tracking and oversight, requirements management, configuration management, quality assurance, subcontract management), which form the basis for project management and establish the context in which technical methods are applied; work products (models, documents, data, reports, forms, etc.) are produced; milestones are established; quality is ensured; and change is properly managed. Software engineering methods provide the technical “how to’s” for building the software, and encompass a broad array of tasks that include requirements analysis, design, program construction, testing, and maintenance. In the last layer described by Pressman we find the software

engineering tools that provide automated or semi-automated support for the process and the methods.

2.2 Problems in software engineering

Software systems are being developed in an increasingly complex business and technological environment (Wynekoop & Russo, 1997, p. 47). The methods and techniques used to develop software keep evolving but the software-related problems also intensify (Pressman, 1997, p. 7). These problems have been called the "software crisis" ever since the late 1960s (e.g. Pressman, 1997, p. 16; Sommerville, 1992, p. 3).

Some basic aspects of these problems are (Pressman, 1997, p. 7):

1. Hardware advances continue to outpace our ability to build software to tap hardware's potential.
2. Our ability to build new programs cannot keep pace with the demand for new programs, nor can we build programs rapidly enough to meet business and market needs.
3. The widespread use of computers has made society increasingly dependent on reliable operation of software. Enormous economic damage and potential human suffering can occur when software fails.
4. We struggle to build computer software that has high reliability and quality.
5. Our ability to support and enhance existing programs is threatened by poor design and inadequate resources.

These problems impose very high demands and pressures on software engineers, as they often must develop large and complex systems with very limited resources. Many development projects fail. However, it is possible to identify a number of common symptoms that characterize these kinds of projects (Booch, 1999, p. 2-5):

- Inaccurate understanding of end-user needs
- Inability to deal with changing requirements
- Modules that do not fit together
- Software that is hard to maintain or extend
- Late discovery of serious project flaws
- Poor software quality
- Unacceptable software performance
- Team members in each other's way, making it impossible to reconstruct who changed what, when, where, and why
- An untrustworthy build-and-release process

When these symptoms appear in a project, they indicate that something in the development process has gone wrong. Different projects fail in different ways, but it appears that most of them fail because of a combination of the following root causes (Booch, 1999, p. 5):

- Ad hoc requirements management
- Ambiguous and imprecise communication
- Brittle architectures
- Overwhelming complexity
- Undetected inconsistencies in requirements, designs, and implementations
- Insufficient testing
- Subjective project status assessment
- Failure to attack risk
- Uncontrolled change propagation
- Insufficient automation

The quest for a solution to these problems has been described as a search for a silver bullet (Brooks, 1999) to slay the monster of missed schedules, blown budgets, and flawed products. According to Brooks (1999, p. 11): "there is no single development, in either technology or in management technique, that by itself promises even one order-of-magnitude improvement in productivity, in reliability, in simplicity". Later research has been more positive and indicated that some techniques, frameworks, and methods can help to solve this even if none can state to have found the silver bullet (e.g. Harel, 1999).

2.3 Approaches for systems development

Textbooks on software engineering (e.g. Pressman, 1997; Sommerville, 1992) along with a large number of journal and conference articles provide thousands of concepts, principles, techniques, approaches, and methods within the field. These are meant to solve problems and improve the development or maintenance of software systems. One of the most important concepts is that of software process models, which provide the framework for which methods are used in or based on. First, software process models will be discussed. Then, since high hopes for improvements are placed on the use of formalized[2] methods, they will get a lengthier discussion.

2.3.1 Software process models

A high level strategy to systems development is often referred to as a software process model, software engineering paradigm, or software life cycle. Such a model is usually divided into a number of phases of different development activities rather than doing everything at once. This aids project planning, guides developers, and provides a good reference point for methods confined to distinct phases (Pressman, 1997). There are many different software process models and variations thereof. In this work only

[2] The term 'formalized' is used to denote formally defined and documented, brand-named or published development methods, of which there are many examples in the literature, rather than ad-hoc approach to systems development (adapted from Fitzgerald, 1996).

three, well known, such models will be described: the waterfall model, the prototyping model, and the incremental model. A few other models will be mentioned without giving any detailed description.

The oldest and most widely used process model is the linear sequential model or the "waterfall model", which suggests a systematic, sequential approach to software development that begins at the system level and progresses through analysis, design, coding, testing, and maintenance (Pressman, 1997, p. 33). The model assumes that all activities of one phase are completed before the project advances to the next phase. This is a simple and understandable model but has some drawbacks. The problems associated with this paradigm are (Pressman, 1997, p. 33):

1. Real projects rarely follow the sequential flow that the model proposes, and changes cause confusion in the project.
2. It is often difficult for the customer to state all requirements explicitly. The model requires this or the outcome will be very uncertain.
3. The customer must have patience. A working version of the program(s) will not be available until late in the project time-span. A major blunder, if undetected until the working program is reviewed, can be disastrous.
4. Developers are often unnecessary delayed when waiting for others to complete dependent tasks.

In the prototyping paradigm, the known objectives and requirements are identified, a quick design is made, focusing on aspects visible to the customer/user, and a prototype (mock-up) of the system developed (Pressman, 1997, p. 35). The prototype is then evaluated by the customer/user and used to refine the requirements for the system. This is iterated to tune the prototype to satisfy the needs of the customer and give the developer a better understanding of what needs to be done. This paradigm is effective in order to define requirements and the prototype should then be thrown away (at least in part) and the system developed in a proper way with an eye towards quality and maintainability. The problem with this paradigm is that the prototype is often used as is and not thrown away. The prototype may have a bad design and inefficient algorithms since the development process is rushed. This causes the system to include less-than-ideal choices as an integral part, which lessens the quality and maintainability of the system (Pressman, 1997, p. 37).

The incremental (also called iterative) life cycle model combines elements of the waterfall model with the iterative philosophy of prototyping (Pressman, 1997, p. 40). The model applies linear sequences of analysis, design, code, and test one after the other (but can be partially parallel, e.g. analysis in second increment while designing the first increment). Each linear sequence produces a deliverable "increment" of the software, usually first a core product and later increments modify or add features and functionality. Such an incremental approach has some advantages over the traditional waterfall process (Kruchten, 1999, p. 73). Those are that risks are mitigated earlier; change is more manageable; there is a higher level of reuse; the project team can learn along the way; and the product has better overall quality.

Some other process models are for example the spiral model, which is an evolutionary software process model that couples the iterative nature of prototyping with the controlled and systematic aspects of the linear sequential model (Pressman, 1997, p. 42). The spiral is divided into task regions (phases), e.g. customer communication,

planning, risk analysis, engineering, construction and release, and customer evaluation. This is similar to the incremental model and has the same benefits. Then there is the component assembly model, which is similar to the spiral model, but object-oriented development is used instead of more conventional methods. The development and use of reusable components is considered to reduce the development time and cost of projects. Last mentioned is the formal methods model, which is used to make a mathematical specification of computer software (Pressman, 1997, p. 48; Bowen & Hinchey, 1999).

2.3.2 Terminology discussion – method versus methodology

It seems that the terms method and methodology are often used interchangeably in the literature (e.g. Avison, 1998) even if there does not seem to be a consensus of their meaning and difference. Therefore a discussion on the terminology is needed.

> "Method is an approach to perform a systems development project, based on a specific way of thinking, consisting of directions and rules, structured in a systematic way in development activities with corresponding development products" (Brinkkemper, 1996, p. 275-276).

The term methodology is described in a dictionary as "a system of methods, principles, and rules, as those of an art or science" (Random House, 1984). When dealing with the development of information systems it is usually implicit in the discussion that the methodology is formalized.

One must by careful not to get confused by the term 'formalized' and think that it only applies to so-called formal methods. Formal methods are methods that have a mathematical basis and are often only used for specification and design of information systems (Bowen and Hinchey, 1999). They are considered in this dissertation to be just one example of methods for systems development. The most important meaning of the term formalized is that the method is documented in a structured way, which is a necessary basis for making the method readily available to its users, e.g. in a book, manual, or electronic documentation.

According to Russo et al. (1996, p. 387): "a system development methodology is a systematic approach to conducting at least one complete phase (e.g. analysis, design or testing) of computer information system development". Additionally, "A methodology consists of a set of guidelines, activities, techniques and tools, based on a particular philosophy of system development and understanding of the target system" (Russo & Wynekoop, 1995, in Russo et al., 1996, p. 387).

According to Brinkkemper (1996, p. 276):

> "The methodology of information systems development is the systematic description, explanation and evaluation of all aspects of methodical information systems development. This definition implies that we restrict the term methodology to scientific theory building about methodical information systems development. The misuse of the term methodology standing for method is a sign of the immaturity of our field, and should consequently be abandoned."

In this dissertation a broad meaning of the term method will be used. The first definition of the term in this section (Brinkkemper, 1996, p. 275-276) is the preferable one, and considered to include documented software processes (commercial or in-

house). In accordance, that which is described in literature as a method, methodology, systems development methodology (SDM), or information systems development method (ISDM) or complies with the abovementioned definition will be termed a *method* in this dissertation. One could go into a lengthy discussion of what is and is not a method but this is avoided by using this broad definition.

2.3.3 Methods

Development methods have formed one of the central topics in information systems and software engineering (Iivari & Huisman, 2001). Many researchers see the solution to the software crisis (see 2.2) in terms of increased control and more widespread adoption of development methods (Fitzgerald, 1996). As discussed in sub-section 2.3.2, a "method is an approach to perform a systems development project, based on a specific way of thinking, consisting of directions and rules, structured in a systematic way in development activities with corresponding development products" (Brinkkemper, 1996, p. 275-276). Different methods are therefore the proposed solutions to the problems encountered when developing software systems.

It has been estimated that there are more than a thousand published development methods (Jayaratna, 1994, in Fitzgerald, 1996). Many of these methods are created by academics and published in the form of an article (e.g. Multiview2 (Avison et al., 1998)) or book (e.g. OMT (Rumbaugh et al., 1991)). Others can be documented in-house methods in organisations (e.g. Cork Organisational Standard Software Process (OSSP) (Fitzgerald, Russo & O'Kane, 2001)). Those are often only used by the organisation itself, but can be used in cooperation between different organisations. There also exist methods that have become or been created as commercial products. Example of this is the Rational Unified Process®[3] (RUP) (Kruchten, 1999).

Some development methods have become a governmental or industrial standard (Fitzgerald, 1996). Examples of this are the Structured Systems Analysis and Design Method (SSADM) (UK, Ireland, Malta, Hong Kong, Israel), Dafne (Italy) Merise (France), NIAM (Netherlands), and Department of Defence Std. 2167 (US). Many major organisations and governments mandate the use of those methods in projects done on their behalf, and contractors that wish to develop software systems for them are therefore forced to use those methods.

To rigorously follow a method will in this dissertation be considered to follow all stages and steps of a method prescribed and considered necessary by the author of the method. When stages or steps are skipped, modified, or blended from different methods, it will be considered an adoption of the method.

Methods can be considered to be composed of coherent pieces termed method fragments (Brinkkemper, 1996). Therefore, methods can either be used in their whole or some method fragments used to construct a situational method tuned to the situation of the project at hand (Brinkkemper, 1996).

It is outside the scope of this dissertation to list and describe the different development methods that exist. Descriptions of them can be found in literature (e.g. Hares, 1994; Kruchten, 1999; Rumbaugh et al., 1991).

[3] Rational, Rational Unified Process, and RUP are trademarks or registered trademarks of Rational Software Corporation in the United States and/or other countries.

2.3.4 Arguments for and against the use of methods

There are many arguments for the use of methods as well as against. The following arguments are summarized from Fitzgerald (1996). First the arguments are summarized in a table (Table 1), which provides a quick overview of the arguments for and against the use of methods for systems development. The table shows two columns. The first column summarizes arguments that support the use of methods and the second column arguments against. The arguments are then discussed in more detail in text.

Table 1 Arguments for and against use of methods (adapted from Fitzgerald, 1996)

Arguments supporting the use of methods	Arguments against the use of methods

• Conceptual underpinnings
  o Reductionistic subdivision of complex development process
  o Facilitation of project management and control, thus minimising risk and uncertainty
  o Purposeful framework for application of techniques and resources
  o Economic benefits with skill specialisation and division of labour
  o Structural framework for the acquisition and systematisation of knowledge
  o Standardisation of the development process, which facilitates interchangeability among developers and increases productivity and quality
• Pressures for increased formalism
  o Desirability of ISO-certification
  o Government development standards (mandated for development projects)
  o Software Capability Evaluation (SCE) programme from Software Engineering Institute (SEI)
  o Literature bias
	• Terminology confusion
	  o Terms misused or confused
	  o Fundamental or artificial differences of terms
	• Generalisation without adequate conceptual and empirical foundation
	• Inadequacies of rational scientific paradigm
	• Goal displacement
	  o Slavish and blind adherence to methods while losing sight of the fact that development of an actual system is the real objective
	• Assumption that methods are universally applicable
	• Inadequate recognition of developer-embodied factors
	• Pressures for new approaches to systems development
	  o Changing nature of business environment
	  o Altered profile of systems development environment
	  o Rapid application development

Arguments for the use of methods

Methods provide a *reductionistic subdivision* of complex development processes, e.g. stages, phases (Fitzgerald, 1996). The 'divide and conquer' principle has been successful in SE as in other sciences to help solve complex problems. Methods thus conceptually divide *what* a system must do and *how* it should do it. Also, the development process is divided into broad categories, e.g. analysis, design, implementation, and maintenance. This in turn *facilitates project management and control* as the development process becomes more visible. This also provides a framework in which steering committees, walkthrough techniques, audit procedures, quality control, inspection practices, and cost and benefit monitoring can be incorporated and used to better control the project and minimize risk and uncertainty.

Methods provide a *purposeful framework for application of techniques and resources* by providing taxonomy of the necessary component activities of development. This in turn gives *economic benefits by allowing skill specialisation and division of labour,* since different skills are needed in different stages of development.

Methods provide a *structural framework for the acquisition and systematisation of knowledge*, which enables transfer of knowledge from skilled and knowledgeable developers to those less skilled, effectively shortening the learning curve for the latter. This formalism in turn allows *standardisation of the development process*, which facilitates interchangeability among developers and increases productivity and quality.

Pressures for the use of methods

There are a number of very influential pressures in favour of the use of methods (Fitzgerald, 1996). *Desirability of ISO-certification* is one of them, in which the process of the relevant organisation needs to be formalised. Also major institutions and governments mandate the use of *development standards* for their development projects, so contractors developing those systems are forces to use them. The *Software Capability Evaluation* (SCE) programme from the Software Engineering Institute (SEI) also presses for method usage in order to provide organisations with better capabilities to produce quality software in a timely and repeatable fashion.

There has been a bias in literature that irrational doings of practitioners are the main source of systems development problems. Also, that the formalisation of practice by adherence to methods is an appropriate step towards a solution to those problems. This creates further pressure towards the increased use of methods in systems development.

Arguments against the use of formalised systems development methodologies

Systems development methods are attractive and have an intuitive appeal, but a systems development method is not the actual systems development, rather it is a framework for organising the system development process (Fitzgerald, 1996). A large part of a method may therefore go towards justifying the method itself.

Terminology confusion is evident in literature and terms often misused. Methods can have fundamental differences in philosophy, objectives, techniques, and focus, but sometimes the distinction is based on artificial differences (product differentiation, personal ego, and territorial imperative).

Methods are constructed by abstracting practices and techniques from a successful development project, and formalising these into a set of guidelines and procedures to

form a development method. This *generalisation is often without adequate conceptual and empirical foundation*, which leaves the method weak.

There is a *problem of inadequacies of the rational scientific paradigm* underlying in methods. The basic problem is the systems development life cycle, which conceptualises developing as following a linear sequence of phases. This requires a perfect foresight, but research has shown that development in practice is not an orderly systematic phased process, but tends to happen all at once. Also, information about what a system should do is obtained in random order and not in a top down fashion. Furthermore, many methods do not cope well with social and human factors. The underlying paradigms of methods are therefore inadequate in many cases.

One of the most harmful potential implications arising from the use of development methods is that of *goal displacement* (Fitzgerald, 1996). This can lead to slavish and blind adherence to methods while losing sight of the fact that development of an actual system is the real objective.

There is a *tendency to assume that methods are universally applicable.* In practice, developers often omit those aspects of a method that does not seem to suit the contingencies of a situation and development is actually an unstructured, evolutionary process. One cannot therefore rely on a method being universally applicable.

Methods often provide *inadequate recognition of developer-embodied factors*. It is people, and not methods, that actually develop software systems, so the factors of the developers and users have to be better considered by methods.

There are a number of pressures for new approaches to systems development. This is partly due to *changing nature of business environment*, which requires organisations to act more effectively in shorter time frames. Older methods are often oriented towards large-scale development with a long development time, but shorter time scales and more flexibility is needed in today's environment. Another factor is a *changing profile of systems development environment*. A fundamental restructuring is taking place in the software industry as companies are relying far less on in-house development of systems, but are pursuing alternative strategies such as buying package software or outsourcing systems development. Researchers in the area of *rapid application development* have pointed out that change in the nature of the demand for systems call for an accelerated development approach to reflect this. A need for a ten-fold increase has been stipulated, away from the older monolithic development methods, which can have a slow pace of development, treat projects as a long-term investment, loose sight of the original business problem, and cause lost interest and deterioration of moral by the development staff (Fitzgerald, 1996).

2.4 Research on methods

There exist numerous studies on methods and their use. First, some interesting empirical studies of the usage of methods will be discussed. The influence of differences of organisations will be mentioned. Method engineering will be introduced and discussed shortly. An example of method tailoring in an organisation will be described. A framework describing actual development of systems in modern organisations will be introduced. Finally, the need for further research will be discussed.

2.4.1 Empirical studies of method usage

Empirical studies on the use of formalized methods in organisations have shown interesting results. A postal survey sent to 776 individuals in different organisations in the UK (21% response rate) showed that 60% of organisations were not using any documented methods, 14% used commercial methods, 12% used internal (in-house) methods based on commercial ones, and 14% used internal methods (not based on commercial ones) (Fitzgerald, 1998). The findings also showed that only 21% of those not using a formalised method did intend to adopt one and 61% of those using it intended to increase its use. Only 6% of the respondents reported following a commercial method rigorously.

Another survey sent to 460 IS managers in the USA (response rate 20%) showed that only 20% of the responding organisations reported never using any method (Russo et al., 1996). Of those using methods, only 6% of the organisations reported that their method was always used as specified and 60% said that the method was always used, but adapted to fit each project. An additional 27% said that their method was used occasionally and the remaining 7% stated that their method was seldom used. A pie chart for these findings is shown in Figure 2. Furthermore, the four most used methods or types of methods used in organisations were identified as Structured Analysis & Design, Information Engineering, Object-Oriented, and Prototyping.

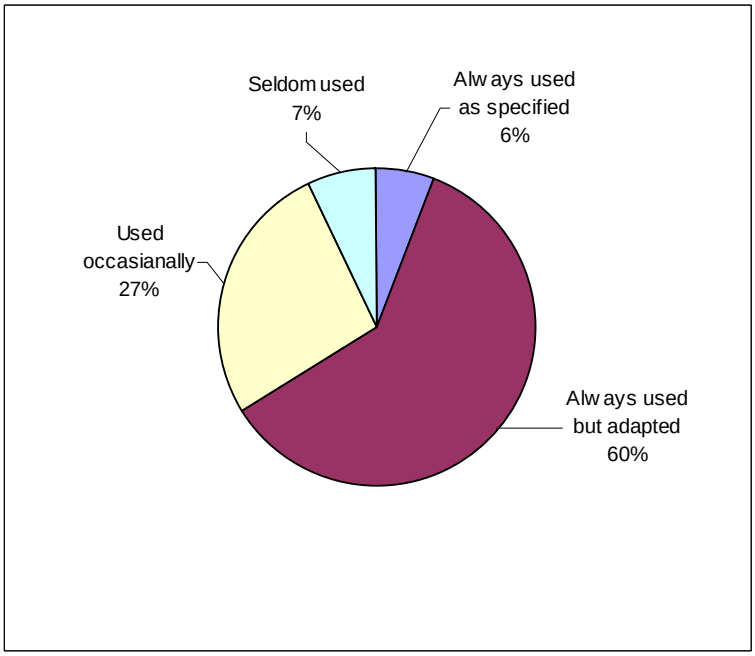

Figure 2 Frequency of method usage (adapted from Russo et al., 1996)

The studies show some difference in the number of organisations using methods, i.e. 40% in UK (Fitzgerald, 1996) and 80% in USA (Russo et al., 1996). Some possible causes of this difference might be that there are differences in how the samples were chosen, how the surveys were done, and that they were done in different countries. This also indicates that one should not believe blindly in absolute numbers from individual studies. One should rather compare different findings to evaluate what might be a good description of reality.

Both studies mentioned above show that only 6% of those using methods use them rigorously as they are prescribed. Method users do not follow all steps and stages as prescribed. While some are followed, others are skipped, and even additional steps can be added (e.g. Bansler & Bødker, 1993; Westrup, 1993, p. 272). This indicates that a majority of method users tailor the methods to meet the specific needs of the organisation and the situation of each project. Other research has supported this and indicated that there is a wide difference between the formalised sequence of steps and stages prescribed by a methodology, and the method-in-action uniquely enacted for each development project (Fitzgerald, 1997). Even when developers are supposed to use a particular method, they use the method when it suits the situation and depart from it when the situation demands a different approach (Power & Richardson, 1996, p. 250).

2.4.2 Organisational differences

Research has shown that organisations with different culture differ in their perceptions concerning the support provided by development methods as well as in their perceptions concerning the impact of methods on the quality of developed systems and the quality and productivity of the systems development process (Iivari & Huisman, 2001, p. 234). Four organisational culture types were distinguished; group culture; developmental culture; rational culture; and hierarchical culture. The clearest differences were that the more hierarchical the organisational culture is perceived to be (with orientation toward security, order, and routinization), the larger parts of development methods are used and the more support they are perceived to provide. This implies that such organisations have a higher chance of getting development methods accepted than organisations with weak hierarchical orientation (Iivari & Huisman, 2001). It was also shown that managers in organisations with high rational culture (focusing on productivity, efficiency, and goal achievement) were more critical towards the deployment of development methods (Iivari & Huisman, 2001).

Development methods are being used in many different types of organisations, but the organisations that do not use a method tend to be smaller, both in terms of overall size and in terms of size of IS staff (Fitzgerald, 1996; Russo et al., 1996).

2.4.3 Method engineering

Method engineering is the engineering discipline to design, construct, and adapt methods, techniques and tools for the development of information systems (Brinkkemper, 1996, p. 276). Method engineering proposes solutions to development problems by engineering situational methods tuned to the situation of the project at hand. This requires standardized building blocks and guidelines, so-called meta-methods, to assemble these building blocks. The building blocks are often called *method fragments* but other terms exist (e.g. method components, method chunks (Ralyté & Rolland, 2001)).

Many researchers in the field of method engineering have proposed frameworks, tools, or languages to base the method assembly process on (e.g. Brinkkemper, 1996; Brinkkemper et al., 1999; Ralyté & Rolland, 2001). Often such proposals are developed without empirical justifications or evaluation outside the research group (Tolvanen et al., 1997). This calls for empirical studies for obtaining a comprehensive view of method engineering in practice.

Potential problems with this approach is that a repository, which can store method fragments is needed, and that could be problematic to implement (Fitzgerald et al., 2001). Also, the role of method engineer will be needed, and recruiting skilled staff can be difficult and expensive.

2.4.4 Example of method tailoring in an organisation

Even if it might seem that developers depart from method usage in an ad hoc way, there are examples of when methods are tailored in a very structured and precise way. An example of this is the method tailoring used at Motorola (Fitzgerald et al., 2001). The organisation has explicitly documented their fundamental software process, the Cork Organisational Standard Software Process (OSSP), which is then tailored precisely to the development process for each project and followed rigorously. The tailoring is done at three levels: a broad Industry level; a more specific Organisational level; and the individual Project level.

The method components at the Industry level are taken from the public domain, known software standards and models (IEEE 1074, CMM, and V-SLCM). At the Organisational level, a number of software processes specific to the various divisions within the organisation are added to the OSSP. The OSSP is meant to cover all aspects that are needed for the software process and the software life cycle in the organisation, and is maintained by adding or improving components as needed. Following the construction of the OSSP, the project manager is responsible for tailoring at the Project level. Depending on the specific characteristics and needs of the project, the necessary elements of the OSSP are chosen.

This example shows the case of an organisation that has recognized both the advantages to be gained from using a standardized software development method and the need to provide a method that is tailored to fit the specific requirements of each development project (Fitzgerald et al., 2001). Furthermore, this makes the project method tailoring quicker and easier, since the OSSP includes the macro-level tailoring and only fine-tuning is necessary by the developers, rather than having each individual developer possess a repertoire of methods to choose from.

2.4.5 Framework describing actual systems development

Fitzgerald (1998, 2001) has proposed an empirically grounded framework, which describes actual development of information systems in modern organisations. The framework makes a distinction between an original method as proposed by its creator and the method-in-action as interpreted and enacted by a developer. It is also indicated in the framework that a formalized method, either commercial or in-house (as long as it is documented), may be the basis for the method-in-action, but is not essential. That is, method-in-action is how developers actually and uniquely enact (instantiate) the development of a particular software system. This framework is shown in Figure 3.

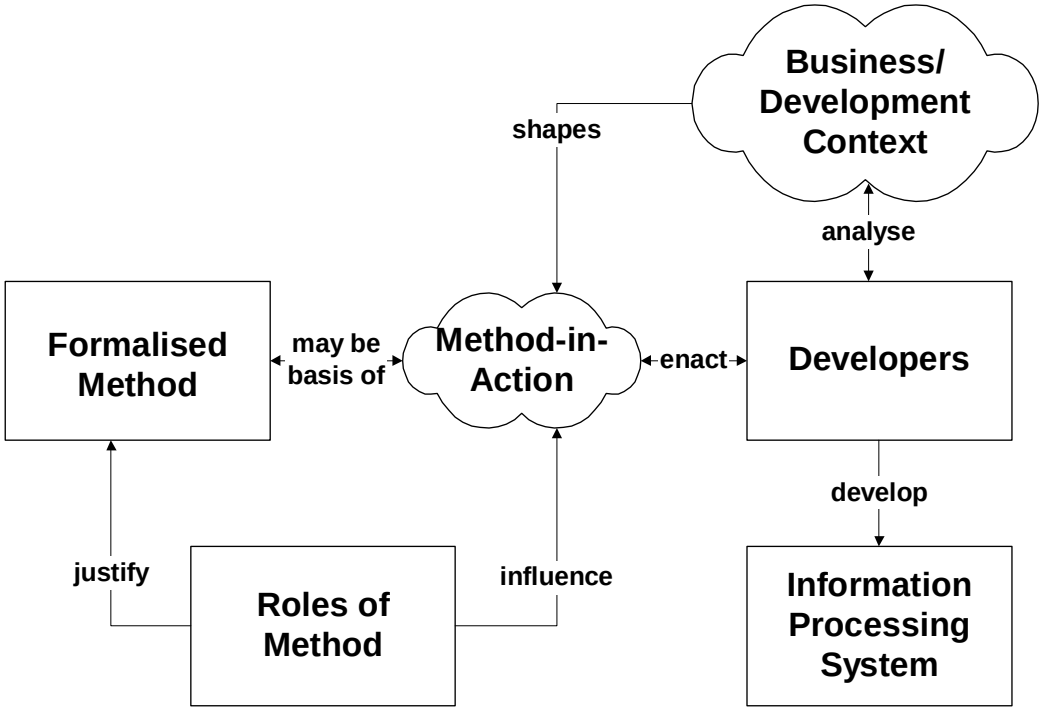

Figure 3 Framework for systems development (adapted from Fitzgerald, 1998, 2001)

The developers have a central role in the framework, since it is people, not methods, who develop systems (Fitzgerald, 1998, 2001). Developer skills, abilities, knowledge, and experience have thus a great effect on how development is enacted. The development environment also has an impact, since organisational size, IS department size, number of developers on project, and project duration, effects whether a formalized method is used or not. The development context, in terms of problem situation and business opportunity is analysed by developers and thus shapes the method-in-action (Fitzgerald, 1998, 2001).

The method plays two sets of roles, rational (overt) and political (covert) (Fitzgerald, 1998, 2001). First, as the rational role, methods facilitate project management by providing transparency into the development process. There are milestones, deliverables, and reviews that help to minimize risks. Methods also provide economic specialisation and division of labour by reducing the development process into series of individual phases of different tasks. A standardisation of knowledge and provision of templates is also provided by the methods, which facilitates inter-communication and inter-changeability among developers, which in turn reduces complexity. Second, as the political role, the methods play a role as a *comfort factor*, possibly giving an illusion, of that "proper" practices are being followed. Other similar roles that a method plays is that of *legitimacy factor* in which organisations may claim to use a methodto win contracts with government agencies, or to help achieve ISO-certification. The role of a *confidence fact*or helps to justify and support decisions in development and thereby provide management with more confidence. By documenting all steps in the development process, methods can provide an *audit trail*, which illustrates the rationale behind decisions taken at various stages during development. Methods can give the organisation an aura of *professionalism* and to individual method "champions" it provides a *power base* by which they can raise their profile within the IS department and the organisation (Fitzgerald, 1998, 2001).

2.4.6 The need for further research

A recent literature survey has analysed existing research on systems development methods (Wynekoop & Russo, 1997). A two-dimensional framework was used to categorize and evaluate existing research on methods. The former was that of the *research purpose*, which was divided into four categories: Method use; method selection, development, and adaptation; method evaluation; and understand/describe. The latter dimension was that of *research method*, which was divided into nine categories: Normative writings; laboratory research; surveys; field inquiries; case studies; action research; interpretive research; descriptive research; and practice descriptions. Over a hundred research papers were identified and classified according to this framework. Table 2 shows the number of studies in each category (adapted from Wynekoop & Russo, 1997, p. 52)

Table 2 Categories of classified research (adapted from Wynekoop & Russo, 1997, p. 52)

	Research question				
Research type	Selection and adaptation	Evaluation	Use	Understand/ describe	Totals
Normative	0	1	0	63	64
Lab	0	1	0	0	1
Field	3	10	5	0	18
Survey	1	6	9	4	20
Case	3	5	3	0	11
Action	1	4	0	0	5
Practice	0	1	3	0	4
Interpretive	1	1	1	0	3
Descriptive	0	1	0	0	1
Totals	9	30	21	67	127

Even if this research survey is not exhaustive it clearly shows which research dominates and where there is a lack of research. Normative research dominates, accounting for over half of the work that was classified. According to Wynekoop and Russo (1997, p. 51), normative research is mostly concept development, presented as factual or objective accounts without interpretation, based on the author's speculations and opinions and without employment of empirical methods or theoretical grounding. Over a third of the normative research identified were frameworks for the selection, comparison or evaluation of methods (Wynekoop & Russo, 1997, p. 52).

Wynekoop and Russo (1997, p. 53) point out that surveys indicate that, whether organisations develop their own or buy commercial methods, the methods are usually adapted. However, they point out, little is known about the process by which these methods are selected, developed, or adapted. Only nine of the studies reviewed addressed any of these issues. Method selection, development, and adaptation is longitudinal and the rational and methods for doing it are determined largely by context. This implies that field research is the key to understanding this process (Wynekoop & Russo, 1997, p. 53).

According to Wynekoop and Russo (1997) there is a need for field research examining how specific methods are used in practice, and also evaluations of methods in actual contexts. This is mainly due to that understanding of the current development environment is necessary to guide the development and application of methods in the future. Studying the use, adaptation, and evaluation of every method in every possible context is of course not possible, but the accumulation of many such studies can form the basis on which to improve current development processes (Wynekoop & Russo, 1997). Many other researchers have criticised the lack of empirical research of the use on methods (e.g. Baskerville & Stage, 2001; Fitzgerald, 1994, 1996), so that acts to support this claim.

3 Problem description

3.1 Problem description

In the field of software engineering there exist many methods that are supposed to aid and support systems development projects. These methods have been a central topic in information systems and software engineering and a lot of effort spent in developing them. Furthermore, there are factors pressing for their use in organisations. Despite this, their practical usefulness is a controversial issue and relatively little is known about how they are actually used. Some of the existing research on this topic indicates that methods are virtually never followed rigorously, i.e. not every step is followed as prescribed. The methods are rather decomposed and some method fragments selected and even modified to fit the specific situation, project, and organisation. The methods are enacted in a "method-in-action", which describes their actual use. It is also quite possible that method fragments are taken from different methods in order to be tailored for use in the current project.

The use of methods in systems development projects can be problematic. There are thousands of methods, of which many are very similar but some very different. The basic problem is how to use the methods for each specific situation. *First*, which method should be chosen? There exist so many methods to choose from and no one can be an expert in all of them. *Second*, how should the method be used? Often a method lacks good examples of how to actually use them in different situations. *Third*, how should the method be tailored to the specific situation? Many methods do not provide information of how to adapt it or change it. Furthermore, methods are often meant to be used in whole, so little guidance is given if only fragments of them are used in the particular situation, or if they have to be modified or blended with other methods. Researchers in the field of method engineering have provided some suggestions on how to solve these problems, but there seems to be a lack of research on how these problems are actually solved by practitioners in the field. That is what this project intends to investigate.

3.2 Problem delimitation

Even if many aspects of method usage mentioned above are interesting to investigate, some focus is needed to make this study more efficient. The aim and objectives are:

Project aim:

> *To investigate how a widespread development method is implemented and used in an organisational setting.*

Project objectives:

> *Investigate how a widespread development method is implemented in an organisation.*
>
> *Investigate how a widespread development method is used in an organisation.*
>
> *Find out if the method is adapted to the organisation, and if so, which factors have effect on that adaptation.*
>
> *Find out if the method is adapted to each development project, and if so, which factors have effect on that adaptation.*

With the use of the term implemented it is meant how the method is adopted by the organisation and put into use, e.g. training of users and method support. An important part of investigating how the method is used is to identify whether the method is adapted or not. Therefore, that aspect is made more explicit in the objectives.

The motivation for selecting this problem is mainly due to the fact that little material was found during the initial literature analysis that tackled these issues. Furthermore, previous research in this field has identified a lack of research on the topic of selection and adaptation of systems development methods (e.g. Wynekoop & Russo, 1997). This indicates that results from such a study can add valuable knowledge to the field of SE.

The results from this project should be interesting and useful to organisations, which are planning to select, implement, and use methods in their projects. It should also be of interest to practitioners to see how others do this. Furthermore, this type of research should be interesting and useful for researchers in the area of methods and especially to those developing or modifying methods.

3.3 Expected results

The results of this study should provide some answers to how a widespread development method is implemented and used in an organisational setting. The answers should give an example of how this is done in practice in an organisation, and therefore give indications about how this might be done in other organisations.

The researcher expects that the results of this study will mainly be a number of descriptions. These could be:

1. Description of the organisational setting
2. Description of the development method under study
3. Description of how the method fits into the organisation
4. Description of how the method is implemented in the organisation
5. Description of how the method is adapted and used at different levels
    - global organisational level
    - local divisional level
    - project level
    - individual developers level

The researcher finds it appropriate to look at the research problem from different levels and this will be reflected in these descriptions.

It will probably not be possible to generalize the findings from this study alone, but the results combined with other similar research can add up to a better understanding on how methods are used in organisations.

It is expected that previous experiences and knowledge of software engineers have a large effect on how methods are used and possibly adapted to specific problem situations. It is also anticipated that conventions and previous projects in an organisation have a large effect. An interesting aspect will be to see if and how in-house and commercial methods are combined.

4 Approach and methods

This chapter describes the approach and methods employed to investigate the aim and objectives of this study (see 3.2). First, a number of possible approaches to tackle the research problem will be briefly introduced and the selection of the case study research technique is motivated. Restrictions on the project, i.e. where the case study will take place etc, are presented. Then, the overall approach for the study is described. Finally, the methods employed to investigate the objectives are described in some detail.

4.1 Possible research approaches and techniques

The project aim (as stated in 3.2) was *to investigate how a widespread development method is implemented and used in an organisational setting*. The research approach selected for this investigation was therefore to do a qualitative study in an actual organisational setting. A qualitative study was deemed most relevant, since the data available to the investigation was considered unlikely to be expressed in absolute arithmetic terms. Data available was considered more likely to be organisational documentation and interview findings, which are better suited to be analysed in a qualitative way rather than quantitatively. Furthermore, a qualitative study was considered more likely to enable the researcher to investigate more deeply the *how* and *why* questions for this study.

According to Gummesson (1988, p. 11), an academic researcher has limited opportunities to understand what is going on in an organisational setting. The ability of a researcher to carry out a project is intimately tied up with the availability of data and information that can provide a basis for analysis and conclusions (Gummesson, 1988, p. 11). Therefore, to be able to investigate in an organisational setting, at least one organisation that is implementing and using a widespread development method had to be contacted and communication with an informant established. Through the informant it had to be possible to gain access to information about the organisation, its development projects and method usage, and to get help in locating suitable people to interview. Without at least one efficient and benevolent informant, the researcher would be lost in an unfamiliar setting (Gummesson, 1988, p. 31).

Some possible and relevant types of research techniques for this study were field studies, action research, and case study research. These will be discussed below and the selection of case study as a research technique will be motivated.

In field studies, organisational changes are studied based on recall and self-report with no variable manipulation (Wynekoop & Russo, 1997, p. 51). A field study would require the researcher to make contact to and study many organisations. While that could give a more generalizable result than a single case study, it would require much more resources than possible in this project, and therefore was not deemed possible.

When doing action research, the researcher participates in the intervention or action studied (Wynekoop & Russo, 1997, p. 51). In this study it could have involved working in a development project and learning from it in order to develop theories about how methods are implemented and used in organisations. While it is an attractive research technique, it requires that the researcher really works in the project for a longer time period than possible in this project. A case study, that can be described more as taking a snapshot of the reality in a short time frame, is therefore more preferable for this project.

According to Yin (1994, p. 20), a case study strategy is most likely to be appropriate for "how" and "why" questions. A case study involves intensive evaluation of small samples using multiple methods, but without statistical or experimental controls (Wynekoop & Russo, 1997, p. 51). That type of research should more likely give good indications on how practitioners actually solve problems rather than doing some other type of research that does not directly involve practitioners in a real context.

According to Patel and Davidson (1994, p. 54), there are many different ways to gather information in order to answer research questions. To name a few, one can use existing documentation, test results, reports, interviews and surveys. None of these techniques can be said to be better or worse than any other, so one has to choose the ones that suit well to answer the research question with the available time and with the possible resources at hand (Patel & Davidson, 1994, p. 54).

To do a case study in an actual organisational setting was considered the most appropriate research approach for this project. It enabled the researcher to look deeply into an organisation and study how practitioners actually implement and use methods in their development projects. The study did not involve statistical measurements or management of controlled variables, which could have taken up too much of the limited project time.

4.2 Restrictions on the project

This project was undertaken as a part of a Bachelor's of Science degree (B.Sc) in Software Engineering at the University of Skövde. The schedule and demands made by the university restricted how this project was conducted. Those restrictions will not be discussed here, but rather the restrictions made by the researchers background and the organisational setting in which the study took place.

4.2.1 Researcher's background

The researcher has some education in development methods and also some experience in working with one particular development method. In three software projects conducted in cooperation between the University of Skövde and external customers the researcher has used a development method called the Rational Unified Process®[4] (Version 5.1.1.1, 1999).

The researcher's experience of the RUP® method is that it is very extensive and a lot of time goes into reading about it and figuring out how it should be used. The researcher noted that a majority of project members had problems with using it. For example: persons could get lost or drown in all the descriptions of what should be done; it was often not prescribed how to do things; good examples were often not to be found; little support was given in how to choose which parts of the method to use; and little support provided in how to modify the method for specific circumstances.

The three academic projects were rather small (120-240 hours per developer) and with very limited resources (4-14 developers). It was therefore not possible to learn and use everything prescribed by the method and it was hard to choose which parts of the method to use. The method was therefore considered to be overkill for such small projects.

[4] Rational, Rational Unified Process, and RUP are trademarks or registered trademarks of Rational Software Corporation in the United States and/or other countries.

Despite the problems experienced with using the RUP method, the researcher felt that the method would be a good development method given that project members receive good education in the method and receive good support for the method in their development projects. This also created an interest by the researcher in how these problems in implementing, adapting and using RUP are solved by professional organisations.

The knowledge and experience of the researcher of the RUP method was considered to restrict the research to the study of the use of that method. This was mostly because RUP is the method best known by the researcher and therefore considered to make it easier for the study than if the method was unknown to the researcher. RUP is also a widely known and used commercial development method, which makes it more interesting to study than some rarely used unknown method. If another method had been chosen for the study, a lot of time would have gone into familiarizing with the method and understanding its implications. That would have left less time available for the actual study.

4.2.2 Organisational setting

Gaining access to necessary information for a qualitative research project can be problematic (Gummesson, 1988). It was therefore considered important to establish contact with an organisation at an early stage the project, in order to see what type of information it would be possible to obtain for this study.

The organisation contacted was Volvo Information Technology AB (Volvo IT) in Skövde. The main reason for contacting this organisation was that the researcher knew beforehand that they had started to implement, adapt, and use the Rational Unified Process (RUP) and than Volvo IT had made the impression on the researcher of being a professional organisation deploying methods in a professional way (Burman, 2001). Furthermore, Volvo IT has cooperated before in research projects, which made it more likely that they had the will and resources to cooperate in such a project again.

A project proposal was sent to the organisation, stating the research problem and possible cooperation in the research, and Volvo IT responded positively. A supervisor, hereafter called an informant, at the company was designated. The informant has been involved with the implementation and adaptation of RUP in the organisation, and has worked on most of the projects that have used RUP in Skövde.

In the initial informal interviews with the informant it became clearer how methods are used at Volvo IT and what type of information was available to the researcher. This initial information could then be used to define in more detail the methods to reach the objectives of the study.

4.3 Research approach

The research approach was to rely on the informant in the organisation to obtain the necessary information. This was done by informally interviewing the informant and by studying selected documents from the organisation. This provided the necessary information about how the method is implemented and used at an *organisational level*. These studies also provided the necessary knowledge and insight in order to prepare interviews that looked deeper into how methods are used at the *project level* and by *individual developers*.

The researcher expected that the information from the documentation and the interviews together would be sufficient to fulfil all objectives of the study and thereby add knowledge to the field of software engineering.

4.4 Techniques to collect data

Two techniques were used to collect data: document study and interviews. The approach is shown in Figure 4. First a document study was conducted in order to obtain information for the description of the implementation, adaptation, and usage of a development method in an organisational setting. The focus of the document study was mostly on organisational aspects. The document study also provided input for preparing a pilot interview. After the pilot interview, the interview questions were modified a little and interviews with other developers conducted. The interviews were focused on the aspects of individual developers, i.e. how they use and modify the method. The interview findings (including the pilot interview) provided additional information for the description derived from the document study.

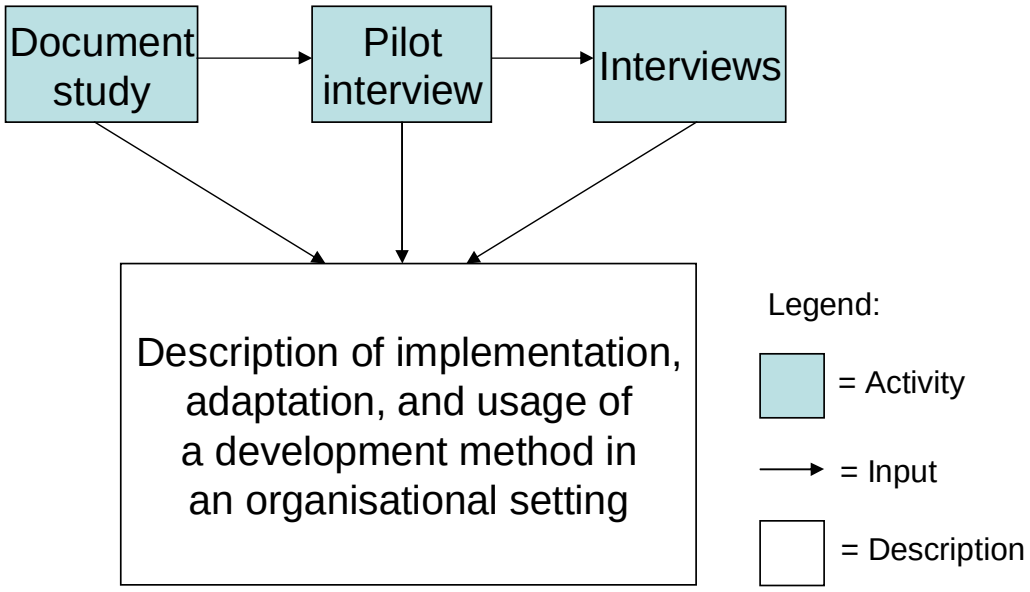

Figure 4 Research approach

For any details that the researcher thought were missing from the derived description, the informant at Volvo IT was contacted for that information.

The informant in the organisation reviewed the descriptions in order to ensure that they gave a precise image of the reality in the organisation, that what the informant had said had not been misunderstood, and only allowable information was described.

4.5 Document study

With help from the informant in the organisation being studied, documentation providing relevant information was identified and selected. The sources for data collection were project documentation, document templates, PowerPoint presentations, and various reports from the organisation. This was used as input for describing the implementation, adaptation, and use of the method as well as describing the company. The results of the document study are presented in section 6.

A selected part of the project documentation was also compared to the document templates prescribed by the method to identify if they were used as described in the method. This enabled the researcher to identify that the method and its documentation (templates) was adapted to the specific situation of the project and helped to prepare questions to use in the interviews.

4.6 Interviews

4.6.1 Interviews as a means to collect data

According to Yin (1994, p. 84) "one of the most important sources of case study information is the interview". It was therefore considered necessary to conduct interviews with developers in the organisation to provide additional information to the results from the document study. The interviews could be more focused on the aspects of individual developers, which was not as easy to do in the document study alone.

When working with questions to obtain information there are two aspects that must be considered (Patel & Davidson, 1994, p. 60). One is the extent to which responsibility is put on the interviewer about the formulation and order of the interview questions. This is called the *level of standardization*. The second is how the questions can be freely interpreted by the respondent (interviewee), based on the person's situation or experience. This is called the *level of structuring*.

Interviews with low level of standardization (or no standardization) are conducted when the interviewee formulates the questions during the interview itself and in an order considered suitable for the respondent (Patel & Davidson, 1994, p. 60). In a fully standardized interview all respondents are asked the same questions in the same order. This is suitable for an interview dependent on what is to be investigated or measured. If the two levels are blended, e.g. an interview in which the most central questions are prepared before the interviews and some successive questions asked spontaneously during the interview, the interview could be called *semi-standardized*.

Level of structuring of interview questions is about the available space to answer questions ("answer room") that a respondent is given (Patel & Davidson, 1994, p. 61). In a fully structured interview, respondents can only select between answers, e.g. "yes" or "no". When the question is not structured there is more room for the respondent to answer freely, e.g. questions like "How do feel about…". Different types of interviews and surveys with different types of applicability are obtained by combining different levels of structuring and standardization of questions (Patel & Davidson, 1994, p. 61).

One type of interview is a *focused* interview, in which a respondent is interviewed for a short period of time (e.g. an hour) (Yin, 1994, p. 84). In such cases, the interviews may be open-ended and assume a conversational manner, but are more likely to be following a certain set of previously derived questions.

An interview technique called "reversed funnel technique" (sw. "omvänd tratt-teknik") is described by Patel and Davidson (1994, p. 65). This technique implies that the interviewer first asks specific questions about different sub-aspects and finally finishes with more general questions. The technique helps respondents to think through the area being discussed and build an opinion at the same time as answering the more specific questions (Patel & Davidson, 1994, p. 65).

The type of interview to be used in a project will depend on the purpose of the interview and whether it is possible to prepare for the interview in advance. If an

interview is conducted without using any special technique for its implementation, the researcher prefers to call those *informal* interviews.

4.6.2 Interviews conducted in the project

Interviews were employed in the project. The informant at the organisation was informally interviewed to get information about the company and about how the method is implemented, used, and adapted. This was done early in the project, in order to get the overall picture of the situation. It was also during these interviews that suitable documents for the document study were identified and then provided by the informant.

The information from the informal interviews and the document study, as well as the researcher's knowledge of the RUP method, provided the necessary knowledge and insight in order to prepare focused interviews. The purpose of the focused interviews was to look deeper into how the method is used and adapted at the project level and by individual developers. The document study did not enable the researcher to look as deeply into this aspect as would have been desirable, so the interviews complemented the document study by providing that information.

The respondents were interviewed for a short time period (about half to one hour). The interview questions were semi-standardized. This provided a previously prepared and more focused agenda for the interviews making better use of interview time and made it easier to compare what respondents said since all had been asked the same central questions. At the same time as most questions were pre-prepared, it was not excluded to spontaneously formulate and ask successive questions during interviews. This was done in order to obtain interesting and possibly important information. Care was taken that this would not take too much time and cause the interview to lose track.

The questions of the interviews were mostly with low level of structuring. Since this study was focused on qualitative analysis rather than quantitative measurements, this was considered more appropriate and more likely to provide the information needed.

In order to ensure as unbiased and easily answered questions as possible, general guidelines for constructing questions were followed. These guidelines were to avoid (Patel & Davidson, 1994, p. 65-66): long questions; leading questions; negations; double-questions; presuming questions; "why" questions that are not successive questions; uncommon words; technical terms; emotive words; ambiguous terms or sentences; and unclear frequency terms (e.g. sometimes, often, regularly).

The questions were written in a numbered list. For most of the major questions, a number of possible and likely successive questions were also written down. These were mostly "how" and "why" questions, and which of them were asked depended on the answer to the previous questions. Spontaneous questions were also asked when it was considered relevant during the interview.

The first interview questions were opening questions about the background of the respondent. These were questions about how long the person had worked at Volvo IT, which had been that persons most major tasks, and which development methods the person had worked with. To start with such general questions was considered a good way to start the dialogue between the interviewer and the respondent. It was also considered important to get this information to see if there were differences in later responses caused by different backgrounds of the respondents.

The later interview questions made use of the “reverse funnel technique” described earlier. The researcher considered it better to start by asking more specific questions about different aspects, rather than to ask directly “how do you use and adapt the Rational Unified Process?” If such a direct question would have been asked directly, the results of the interviews would have been much more uncertain. By asking more specific questions it was ensured that more material would be obtained for the analysis and to enable the respondent to have some time to reflect upon the specific questions and build opinions during the interview before answering more general questions.

The specific questions were restricted to projects using the Rational Unified Process in which the respondent had participated. The first were about which such project the respondent had participated in and which roles the person had played. The next questions were focused on three central aspects of RUP, namely that of roles, activities, and document templates (for artifacts). The questions were about how those are used and how they are modified. Successive questions would provide knowledge of how the person selects parts to use or skip from the method, how parts might be added, and which factors have the greatest effect on this.

Following the specific questions, more general questions were asked. These were about how RUP is related to other methods in use at Volvo IT, how the respondent uses RUP in general terms, and what factors have the greatest effect on how and why RUP is adapted in the organisation. It was expected that many parts of these questions could be already answered during the previous questions, but still these questions provided a chance to mention aspects that were not previously mentioned and summarise earlier discussions in more general terms.

With the help of the informant at Volvo IT, suitable persons to interview were contacted. Four developers were contacted and an interview meeting scheduled. A letter which shortly explained the purpose of the study and the interviews, along with the list of questions, was sent to the respondents beforehand. This text is shown in appendix A.

First a pilot interview was done with the informant, and the results evaluated to which degree they could help to fulfil the objectives of the study. The results from the pilot interview were considered very good and sufficient and only small changes to the questions considered appropriate. After making small modifications to the questions (shown in appendix B), the rest of the interviews were conducted.

The interviews were conducted in a conference room located at Volvo IT. This ensured that the time to participate in the interview would be as little as possible for the respondents. One of the five planned interviews had to be dismissed because the respondent had other urgent matters to attend to. It was not possible to plan another interview in time and the four interviews were considered sufficient for this study.

All respondents accepted that the interview was recorded on magnetic tape with a micro cassette recorder. To keep the material anonymous, each interview was numbered randomly, and the number written on both the tape and written material regarding the interview. A list of the mapping between interview numbers and the persons was kept separately by the researcher. All the respondents allowed the researcher to contact them later regarding the interview if needed for any clarification or such.

Respondents were asked it they were comfortable with talking English during the interview. Half of one interview was conducted in English, but it became apparent

that this obstructed the respondent to some extent in answering questions. Swedish then became the most preferable language of choice during the later half of that interview and in all the other interviews.

After a short discussion at the beginning of the interview, the recorder was started. The questions were asked in the same order as on the question list. Some interesting and important things were noted down on paper. This was done as a failsafe measure, in case the recording failed, and to keep track of which questions had been asked during the interview. Successive questions were asked in relation to each question on the list. These had some differences due to differences in the roles and experiences of the respondents. The interview time varied between 40 and 60 minutes.

All the interviews were summarised in a document. This was done with the help of the recording and the written material from the interviews. The summary omitted irrelevant or repeated statements. This resulted in 19 pages of text.

The summary from the interviews were analysed and described in a number of relevant categories. These categories summarise the results from the interviews and are shown in section 6.3.

The information from documentation and interviews together were considered sufficient for the analysis in order to fulfil the objectives of the study.

5 Introduction of the Rational Unified Process

Since this study is concentrated on the implementation, adaptation, and usage of the Rational Unified Process®[5] as a systems development method, an introduction of RUP® is needed. The promoters of RUP choose to call the method a software engineering process, but since RUP fulfils the definition of a development method used in this study (see section 2.3.2) the terms will be used interchangeably in the text. This introduction will provide only enough detail in order to understand discussions about RUP in later chapters. The material is taken and adapted from Rational (2002) when not stated otherwise.

5.1 Rational Unified Process

The Rational Unified Process® or RUP® product is a software engineering process created and maintained by the Rational Software Corporation. It provides a disciplined approach to assigning tasks and responsibilities within a development organisation. Its goal is to ensure the production of high-quality software that meets the needs of its end users within a predictable schedule and budget.

The RUP product consists of an online version of the Rational Unified Process (a fully hyperlinked Web site description in HTML) and a set of manuals that describe the process.

The Rational Software Corporation also provides a number of tools and workbenches that can be used with the RUP method, but are optional. All these tools and the method are integrated, and are therefore usually used together. These tools will not be described here.

5.2 Software development best practices

The Rational Unified Process is based on and captures many best practices of modern software development in a form that is suitable for a wide range of projects and organisations (Kruchten, 1999, p. 18). This follows the following six best practices:

1. Develop software iteratively
2. Manage requirements
3. Use component-based architectures
4. Visually model software
5. Verify software quality
6. Control changes to software

A detailed discussion of these properties can for example be found in Kruchten (1999).

[5] Rational, Rational Unified Process, and RUP are trademarks or registered trademarks of Rational Software Corporation in the United States and/or other countries.

5.3 RUP architecture

The figure below (Figure 5) shows the overall architecture of RUP®[6]. The horizontal axis represents time and shows the lifecycle aspects of the process, the vertical axis represents disciplines, which group activities logically by nature.

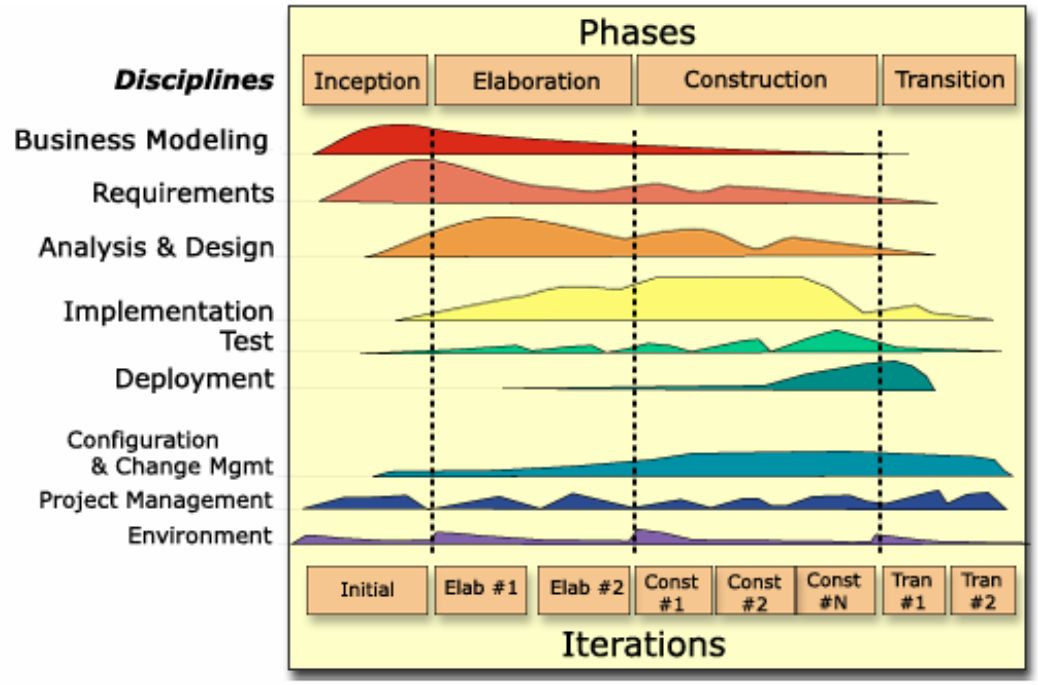

Figure 5 The architecture of RUP® (from Rational, 2002)

The RUP method has two dimensions: The first dimension represents the dynamic aspect of the process as it is enacted, and it is expressed in terms of phases, iterations, and milestones; The second dimension represents the static aspect of the process: how it is described in terms of process components, disciplines, activities, workflows, artifacts, and roles.

The graph in Figure 5 shows how the emphasis varies over time. For example, in early iterations, more time is spend on requirements, and in later iterations more time is spend on implementation.

From a management perspective, the software lifecycle of the Rational Unified Process (RUP) is decomposed over time into four sequential phases, each concluded by a major milestone; each phase is essentially a span of time between two major milestones. At each phase-end an assessment is performed to determine whether the objectives of the phase have been met. A satisfactory assessment allows the project to move to the next phase. The phases are named Inception, Elaboration, Construction, and Transition.

[6] Rational, Rational Unified Process, and RUP are trademarks or registered trademarks of Rational Software Corporation in the United States and/or other countries.

The development in RUP is divided into nine disciplines (core workflows). Each shows the activities you may go through to produce a particular set of artifacts. These disciplines describe at an overview level—a summary of all roles, activities, and artifacts that are involved. They also describe, at a more detailed level, how roles collaborate, and how they use and produce artifacts. The disciplines of RUP are:

- Business Modeling
- Requirements
- Analysis & Design
- Implementation
- Test
- Deployment
- Configuration & Change Management
- Project Management
- Environment

The details of each of these disciplines will not be listed here, and the reader is referred to Rational (2002).

Each iteration consists of planning, requirements, analysis & design, implementation, and testing, in various proportions depending on where the iteration is in the software lifecycle. Early iterations focus on requirements and architectural design, whereas late iterations focus more on design, implementation, and testing. Thus, part of the Iteration Plan is to decide how the various disciplines are to be exercised for each iteration.

One pass through the four phases is a **development cycle**; each pass through the four phases produces a **generation** of the software. Unless the product is no longer needed it will evolve into its next generation by repeating the same sequence of inception, elaboration, construction, and transition phases, but this time with a different emphasis on the various phases. These subsequent cycles are called **evolution cycles.** As the product goes through several cycles, new generations are produced.

Evolution cycles may be triggered by user-suggested enhancements, changes in the user context, changes in the underlying technology, reaction to the competition, and so on. Evolution cycles typically have much shorter Inception and Elaboration phases, since the basic product definition and architecture are determined by prior development cycles. Exceptions to this rule are evolution cycles in which a significant product or architectural redefinition occurs.

5.4 Roles and activities in RUP

In RUP®, a role is an abstract definition of a set of activities performed and artifacts owned (Rational, 2002). Roles are typically realized by an individual, or a set of individuals, working together as a team. A project team member typically fulfils many different roles. It is important to note that roles are not individuals; instead, they describe how individuals behave in the business and what responsibilities these individuals have when playing that specific role.

Roles have a set of cohesive activities that they perform. These activities are closely related and functionally coupled, and are best performed by the same individual. The

activities are closely related to artifacts. Artifacts provide the input and output for the activities, and the mechanism by which information is communicated between activities.

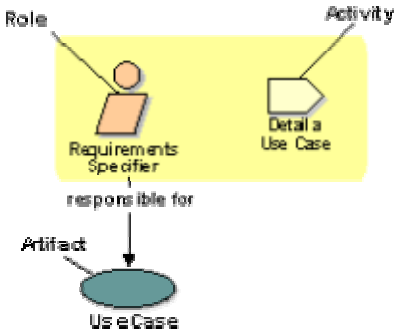

Figure 6 Roles, activities, and artifacts (from Rational, 2002)

Figure 6 shows the relationship between a role (e.g. Requirement Specifier), the activities that role is responsible for (e.g. Detail a Use Case), and the artifact the role is responsible for and which also is the result of the activities (e.g. Use Case).

There are a total of 33 roles prescribed by RUP (in its latest version, i.e. 2002.05.00) (Rational, 2002). These are organised into five sets which will be described shortly below.

The Analyst role set organizes those roles primarily involved in eliciting and investigating requirements. Roles included are:

- System Analyst
- Business Designer
- Business-Model Reviewer
- Business-Process Analyst
- Requirements Reviewer
- Requirements Specifier
- Test Analyst
- User-Interface Designer

The Developer role set organizes those roles primarily involved in designing and developing software. Roles included are:

- Capsule Designer
- Code Reviewer
- Database Designer
- Implementer
- Integrator
- Software Architect
- Architecture Reviewer
- Design Reviewer
- Designer
- Test Designer

The Tester Role Set organizes those roles that deal with the specific skills unique to testing. Note that there are additional roles involved in the Test discipline that build on and extend the base skills of other role sets. These additional roles can be found in the other role sets arranged by the base skill-set they extend (i.e. Manager, Designer, and Analyst). Role included:

- Tester

The Manager role set organizes roles primarily involved in managing and configuring the software engineering process. Roles included are:

- Process Engineer
- Project Manager
- Change Control Manager
- Configuration Manager
- Deployment Manager
- Project Reviewer
- Test Manager

The additional role set organizes those roles that involve support or miscellaneous roles on the project. Roles included are:

- Stakeholder
- Any Role
- Course Developer
- Graphic Artist
- Tool Specialist
- System Administrator
- Technical Writer

The details of each of these roles will not be listed here, and the reader is referred to Rational (2002).

5.5 Artifacts

Artifacts are either final or intermediate work products that are produced and used during a project. Artifacts are used to capture and convey project information. An artifact can be any of the following: A document, such as Business Case or Software Architecture Document; a model, such as the Use-Case Model or the Design Model; a model element; that is, an element within a model, such as a class, or a subsystem

Models and model elements have reports associated with them. A report extracts information about models and model elements from a tool. A report presents an artifact or a set of artifacts. Most artifacts have guidelines, which describes the artifact in more detail.

To make the development of a complete software system manageable, the artifacts are organized into sets corresponding to the disciplines. Several artifacts are used in a number of disciplines; for example, the Risk List, the Software Architecture Document, and the Iteration Plan. These kinds of artifacts belong to the artifact set where they are primarily produced.

Figure 7 shows the most central artifacts in RUP® (from Rational, 2002). The arrows represent the fact that artifacts effect and provide input to other artifacts. This figure shows only a part of the artifacts, for a more detailed description of the artifacts the reader is referred to Rational (2002).

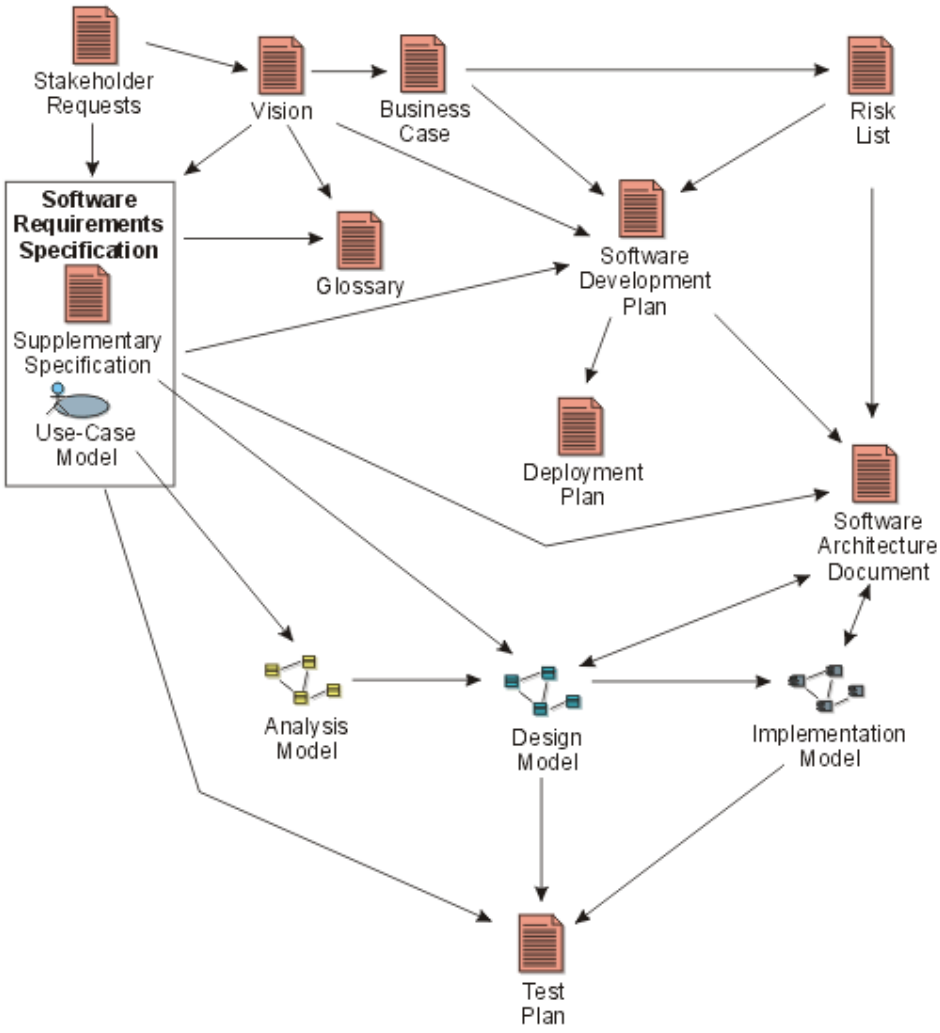

Figure 7 The most central artifacts in RUP® (from Rational, 2002)

5.6 RUP configuration and modification

RUP provides some support in how to configure and adapt the method to the specifics of organisations and projects.

The environment discipline in RUP focuses on the activities necessary to configure the process for a project. It describes the activities required to develop the guidelines in support of a project. The purpose of the environment activities is to provide the software development organisation with the software development environment—both processes and tools—that will support the development team.

RUP provides a number of support documentation. Examples are: The basics about how to configure the Rational Unified Process; a step-by-step procedure of how to implement a process in an organisation (however, this procedure is not described in terms of roles, activities, and artifacts); and an explanation of how to implement a process, together with supporting tools, in a software development project.

The Process Engineer Toolkit in RUP provides tool support for configuring a process. This includes tools and examples for creating organisation and/or project web sites based on the RUP method.

The Development Case artifact (document) in RUP describes the development process that has been chosen to follow in a particular project. The purpose is to capture the tailored process for the individual project. The Development Case is created early in the Inception phase and is updated throughout the project as needed. The Process Engineer role is responsible for this artifact.

A first version of the Development Case is created at the onset of the project. It is recommended that the Development Case is developed in increments, covering more and more of the disciplines in each iteration. The first version of the Development Case will normally only cover a subset of the disciplines. In each of the subsequent iterations, more will be covered by the Development Case. As the results of each iteration are evaluated, the Development Case is likely to change based on the lessons learned.

Normally, a project does not start using all disciplines in RUP. If that is the case, the corresponding sections can be removed from the Development Case document. If needed, more information is added about how to use the artifacts for each discipline. If needed, references are added to guidelines and information that the project wants to use in addition to the RUP method.

The Development Case in this way describes for a particular project what disciplines, roles, activities, and artifacts will be used in the project. This is described for each development phase. The motivations for modifications are also described here.

The Development Case states for each artifact and each development phase a classification of its use and the level of formality of that artifact. The use of an artifact is in four classes: **Must have**, i.e. you must use this artifact. It is a key artifact and may cause problems later in development if its not produced; **Should have**, i.e. you should have this artifact, if at all possible, but it is negotiable. If you do not produce this artifact, you should be able to justify why not; **Could have**, i.e. this artifact doesn't have to be produced. It's only produced if it adds value and if there's enough time; and **won't have**, i.e. you won't use this artifact. This may occur where a RUP artifact is replaced by a local artifact. The level of formalism and ceremony an artifact requires can be in three levels: **Formal-External**, this is the highest level, meaning that the artifact shall be formal and correct enough to be meaningful for external parties such as customers, users, other projects, other organisations etc. This does always require reviewing and approval; **Formal-Internal**, this is the second highest level, meaning that the artifact shall be formal and correct enough to be meaningful for the internal project or organisation. This often requires reviewing; and finally **Informal**, this is the lowest level, meaning that the artifact does not need any formalism at all and does not require reviewing.

The Development Case is not the same as the project plan, which is usually described in the Software Development Plan artifact.

5.7 Evolution of RUP

The Rational Unified Process is constantly evolving. New releases of the product regularly add improvements or additions to the method. It is not in the scope of this study to provide a detailed account on the history of RUP but a few relevant changes to the method will be mentioned.

The researcher made some comparison of RUP from 1999 (version 5.1.1) and the newest version (Rational, 2002). From this comparison it was clear that many improvements and additions have been made to the method. The architecture of the web presentation has been changed in order to make it easier to find relevant information, and is therefore easier to use. The number of roles and their responsibilities has been changed a little. The number of roles was previously 30 and is now 33. The roles are now organised into five sets, making it easier to see which roles are related and logically connected. Descriptions of the process have been improved and more examples added. The new version includes much more support for configuring and modifying the method, especially for small projects. The researcher therefore considers the method to have improved a lot with time and become easier to use. The method has undergone many more changes and additions, but it was not considered relevant to describe them here. The evolution of related tools was not looked into.

6 The case study

This chapter describes the results from the case study. First a description of the organisational setting is given. Then the results from the document study and the interviews are presented.

6.1 Organisational setting

The organisational setting of this study is that of Volvo IT in Skövde. Volvo IT is a large organisation positioned at many places in Sweden and many other countries. This section will provide a short description of Volvo IT globally, in Skövde, and in Göteborg.

6.1.1 Volvo IT globally

Volvo Information Technology was first established in 1966 and then re-established in 1998 as an independent organisation, and is Volvo's own information technology company. The company has global responsibility for IT solutions in the Volvo Group and Volvo Cars. In recent years, they also provide some solutions for a number of external customers. Volvo IT provides a uniform and consistent information technology infrastructure and systems development services throughout Volvo companies worldwide, which allows Volvo companies to share information and to centralize some investments in hardware, software, and application development resources (Volvo Information Technology North America, 2002).

Volvo IT employs about 4300 people (including contractors) in many locations worldwide (Figure 8). Most of the employees are at various locations in Sweden (about 2650).

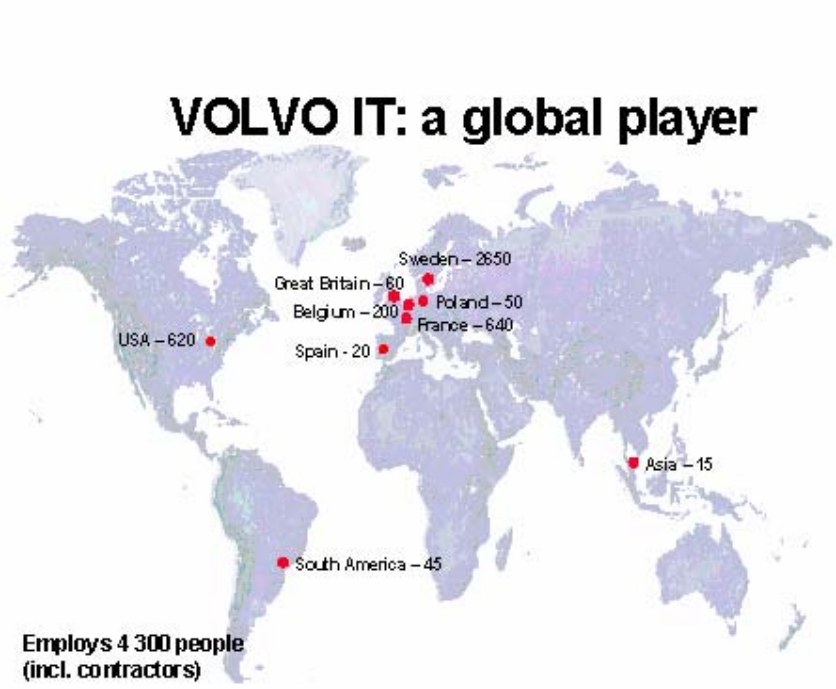

Figure 8 Global map of Volvo IT locations (from Volvo IT, 2002)

The Volvo Group, the biggest internal customer of Volvo IT, is a corporation that is intensively focused on transport solutions for commercial use (Volvo Group, 2002). The business areas of the Volvo Group are trucks, buses, construction equipment, marine & industrial power systems, aero, and financial services (Figure 9 (Volvo IT, 2002)). Consequently, there are many different types of computer systems that Volvo IT has to manage (e.g. mainframe operations, technology systems, commercial systems, business applications, production systems) using different platforms (e.g. OS/390, OS/400, UNIX, VMS, and Windows NT) (Volvo IT, 2002).

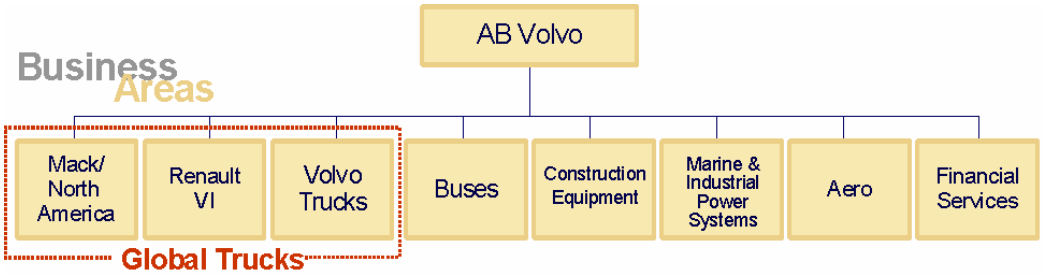

Figure 9 Volvo Group business area map (from Volvo IT, 2002)

The vision of Volvo IT is to be a highly successful IT company which contributes to its customers' profitability using skilled and committed employees (Volvo IT, 2002). The company's mission is to create added business value for its customers by supplying cost-effective, complete IT systems and solutions. Volvo IT highly values that these systems are cost effective, that they fulfil requirements, are delivered on time, and have high quality. For this Volvo IT plans to provide fast and reliable service to its customers by being close to the customer, employee well-qualified staff, collaborate with the best suppliers, and continuously invest in expertise development and extend the customer base.

6.1.2 Volvo IT in Skövde

About 136 people work at Volvo IT in Skövde. The organisation is divided into a number of departments, which are shown in figure 10.

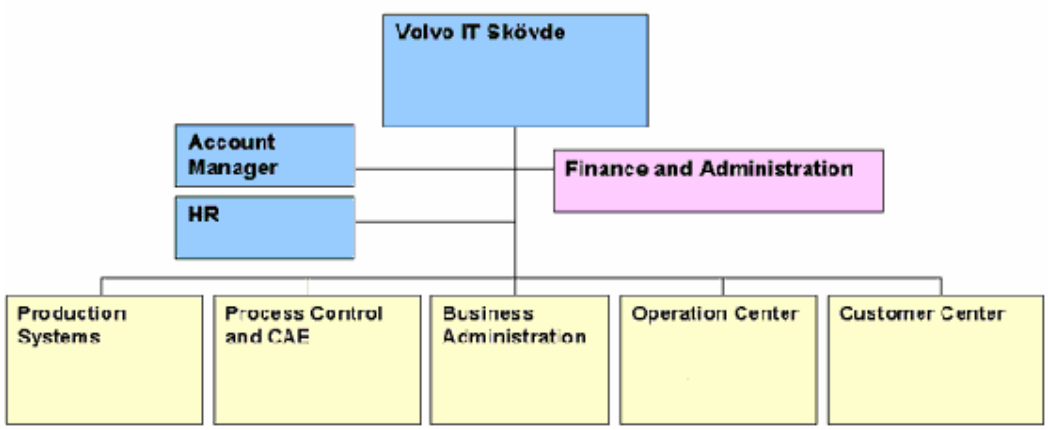

Figure 10 Volvo IT in Skövde (adapted from Volvo IT intranet)

The Production Systems department develops and administers administrative systems solutions for the whole production chain from order-/prognosis management to delivery. These systems usually link together applications that are near the production floor, so close contact and collaboration is with the Process Control and CAE department. The department works closely with their clients with the analysis, design, and implementation of both commercial and in-house systems.

The Process Control & CAE department is composed of two teams. The Process Systems team works with systems and applications that are near the production floor. This team focuses mostly on System Project Management and Technical Systems work in connection with production systems bought by their clients from external contractors. Examples are AGV-systems, Crane warehouses, Progress and Material Report systems, Engine Test systems, and Quality Assurance systems. The team also supports and maintains existing systems and develops new systems. The CAE-team investigates for their clients the best and most effective way to use computer support in the development of new products and production processes. This involves "virtual" tools and simulations.

The Operation Centre department focuses on Techniques, Method and Systems Development support, and Enterprise Evolution. The department is divided into three teams: Service (sw. Drift), which is responsible for monitoring and operational service of the computer systems in Volvo IT; SU-Support, which is responsible for administration, development, and enhancement of methods and techniques in systems development and production environments at Volvo IT in Skövde; and Technique, which is responsible for infrastructure (networks and telecommunication), server operation, and database management.

The Business Administration department mainly deals with development and administration of administrative systems. This group has competence in various development tools and methods, Project Management, Investigations, and Office products.

The Customer Center department provides local IT support for customers. One team of the department goes to the customer's workplaces and provide support on-site, mostly general user support. Another team takes care of technical support, like installing software or hardware and drawing cables and such.

As can be seen from the descriptions of these departments at Volvo IT in Skövde, the organisation has to develop, configure, administrate, and maintain a number of different systems. Most of the clients have been in-house, various departments of Volvo at Skövde, but the company has had some external clients as well in recent years.

At Volvo IT there are a number of methods for problem investigation, systems development, and systems maintenance in use. The central method that has been in use for application development in Sweden is an in-house method called the AU-model. This method is currently being replaced by the Rational Unified Process (the method is described in chapter 5). The AU-model will be described later in this chapter (in 6.2.1).

6.1.3 Volvo IT in Göteborg

Volvo IT has its headquarters in Göteborg, where there is a department named Common Skills and Methods, which is responsible for evaluating, introducing, and supporting the use of development methods in all departments of the company. The

staff of this department usually work at least 50% in other departments as well. This is done in order to ensure that they gain experience in using the methods that they are responsible of supporting or introducing in the rest of the organisation.

The Common Skills and Methods department provides method descriptions and licences to the other departments. It also provides guidance for adapting and implementing the methods. Most of the information is made available on the Internet on a method support web site located in Göteborg. Only employees of Volvo IT and external consults working for Volvo IT have access to this information.

Some of the work of method support is distributed to other departments, but it is managed from the Common Skills and Methods department.

6.2 Results from the document study

First a short description of the AU-model currently in parallel use at Volvo IT in Sweden will be given. Then the reasons for choosing RUP as the central development method at Volvo IT will be discussed. This follows the descriptions of how RUP fits into Volvo IT and how RUP is implemented in the organisation. How RUP is adapted and used at different levels in the organisation is then described and exemplified.

6.2.1 Short description of the AU-model

The AU-model is a development method created by the Volvo IT organisation in Sweden. AU stands for "Administrativ Utveckling", which means administrative development. The model is based on the "waterfall model" and provides a description of *what* should be done in different phases of development, but provides little guidance of *how* to do it. It is not an iterative model.

The AU-model is divided into three parts: Development model (sw. utvecklingsmodellen); Project management model (sw. styrningsmodellen); and Maintenance and Enhancement model (sw. förvaltningsmodellen) (Volvo Data AB, 1991). Each part has its own purpose, but all are needed for successful results. The model is furthermore divided into three phases; Preliminary study (sw. förstudie), which for example analyses enterprise problems and sets goals and ambitions; Project Planning (sw. projektering), which for example describes solutions and changes, and concretizes goals into a project management plan; and Realization (sw. genomförande), in which the actual systems development takes place.

One of the major drawbacks of using the AU-model is that only some departments of Volvo IT in Sweden use it and no other organisation are not using it. This makes the sharing of resources and cooperation between departments of Volvo IT as well as other organisations more cumbersome and difficult than if a mutual development method was used.

The AU-model is described in a number of manuals (in Swedish). These are internal documentation of Volvo IT in Sweden and are therefore not public. A more detailed description of the model is to be found in Sandström (2001).

6.2.2 The reasons for choosing RUP as the central development method

There is a range of development methods in use at Volvo IT. Most of these methods are internally developed (in-house methods) but some are commercial methods.

Volvo IT made a decision in 1999 that the Rational Unified Process®[7] was to be their central method in all new application development.

The basic problems that called for a new development process at Volvo IT were that:

- The operations of Volvo are more and more rapidly changing
- There is a demand for a global and common way of work
    - Different departments have used different methods
- The development process must be modifiable
- IT-projects must be able to handle changes during development
- The development process must enable planning, control, and follow-up of a project in an efficient way

The strategy for solving these problems was to:

- Establish a common process for application development within Volvo IT
- Integrate Business Engineering and Application Development
- Replace the old development methods with state-of-the-art methods and tools supported globally

The solution chosen by Volvo IT was to implement a new development method in the whole organisation. This is based on two main parts, which is the Project Control Model (PCM) developed in-house, and the Rational Unified Process® (RUP®) from Rational® Software. Some of the expected effects of introducing PCM and RUP are (Volvo IT, 2001b):

- More projects on time
- Shorter lead time until delivery of first increment
- Better product fit to actual needs at time of delivery
- Reduced rework costs
- Better product maintainability

RUP was considered to provide good support for the needs of the organisation for the central development method. The PCM model had been introduced earlier into the organisation and was considered to give better support for Project Administration and Management. The two methods are therefore used together.

6.2.3 How RUP fits into Volvo IT

A map of how different sets of methods work together in systems development at Volvo IT is shown in Figure 11. RUP is the Application Development Process that provides both the framework and methods for Requirements & Analysis, Software Architecture, Design, Implementation, and Deployment. The sets of methods are modified according the development environment and platform (indicated by the grey box). RUP is in this way supposed to be the glue that holds together the development process and all the other different methods that are in use in Volvo IT.

[7] Rational, Rational Unified Process, and RUP are trademarks or registered trademarks of Rational Software Corporation in the United States and/or other countries.

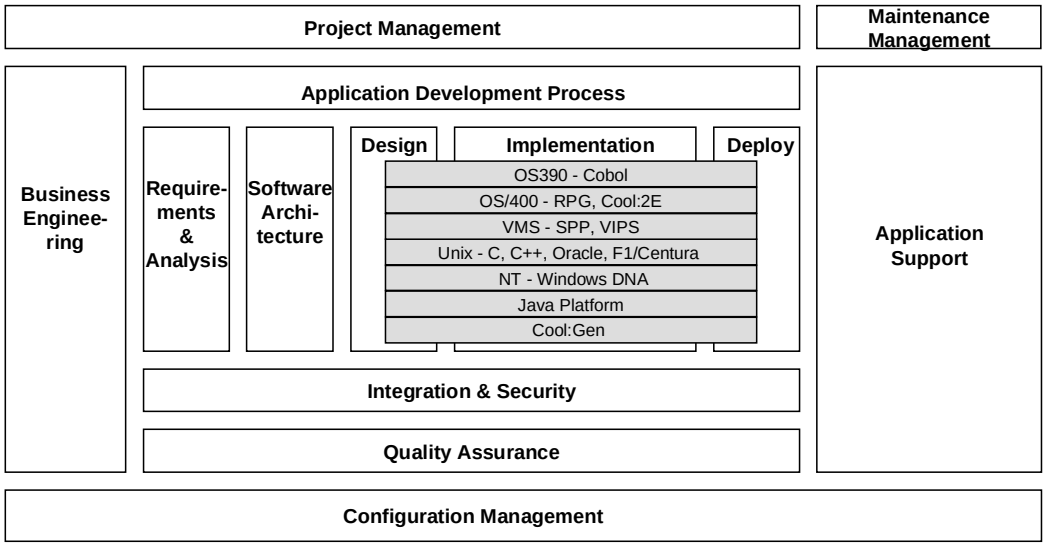

Figure 11 Volvo IT Method Area Map (from Volvo IT, 2002b)

Volvo IT has developed its own methods for project management, project administration, maintenance management, application support, integration & security, quality assurance, and configuration management. For business engineering, each project can choose between using an in-house method from Volvo IT or using RUP. The rationale for keeping these older in-house methods in use are that Volvo IT considers them to give better support for those aspects of systems development than RUP. Another reason is that their staff has competence in using the older methods while it would be unadvisable to exchange all methods at the same time.

RUP is adapted to the development environment it is used in. The grey area in Figure 11 indicates this. This adaptation mostly results in guidelines in how to use the method in that particular environment. The adaptation is typically done by the department that has the best competence in using that environment but it is managed from the Common Skills and Methods department. The guidelines are made available to all departments though the method support website.

The Project Control Model (PCM) is the method that is used on top of RUP for project management. The method provides more support for project management and reporting to higher-level administration than RUP does. Some of the activities and documentation prescribed by RUP for the Project Manager are therefore replaced by parts of that method. A mapping of which activities and documents are to be replaced between the methods has been made by the organisation. This mapping avoids double job in the documentation. It is mainly the Project Manager that is affected by this method; other developers in RUP projects do not have to concern themselves with it since the RUP documentation is sufficient for them. A special Project Manager Programme training program is in place to educate and train Project Managers in management, the use of PCM, and in RUP.

The use of the PCM method is mandatory in all projects at Volvo IT, both application development and other development. This is due to an ISO 9000 standardisation that the organisation has achieved by using that method.

The Maintenance Control Model (MCM) is another in-house method, which is used. The method provides maintenance aspects that lack support in RUP.

A method called CAMP is used for pre studies of organisational problems, investigations of different types, and business modelling. RUP only supports business modelling, but not the other types of investigations, so this method will continue to be in use at Volvo IT. It is quite common that the result from a CAMP investigation becomes the input for the start of a RUP project.

Other methods, descriptions, and guidelines that have been documented at Volvo IT and do not have a similar support in RUP are still in use in the organisation.

6.2.4 How RUP is implemented at Volvo IT

The experiences from six small pilot projects using RUP in 1999 showed that introducing a new development method is a large investment in building competence (Volvo IT, 2001b). Following the recommendation from the Common Skills and Methods department, Top management decided on a "project by project" implementation strategy where RUP is introduced incrementally into the organisation. To be able to support the projects, they started training people to become "RUP coaches" and "RUP specialists". In the first projects the specialist role was filled with external consultants, but as experience grew, they were replaced with in-house staff.

Suitable projects were chosen to be implemented using RUP. Suitable projects were in the beginning considered in-house or enhancement application development projects that required a project team of 3-10 persons, had duration of 3-9 months, and an effort of 2000 – 3000 work hours. Since the first RUP projects were considered to be successful and take no more time than with the older development methods, the implementation of RUP projects proceeds, and are preferably done for external customers. The current rule of thumb at Volvo IT for RUP projects is that they should be new and small application development projects, effort of 1000 – 3000 work hours, with 3-5 persons. Furthermore 1-2 of the developers should be experienced with RUP (preferably coaches) and 2-3 of the persons be learning RUP during the project. To have so small projects implementing RUP in the beginning is considered to be suitable by the organisation. The plan is to incrementally implement the method in the organisation, so when about half of the developers in the organisation are experienced with RUP more of bigger projects will implement the method as well and in the end most application development will be done using RUP.

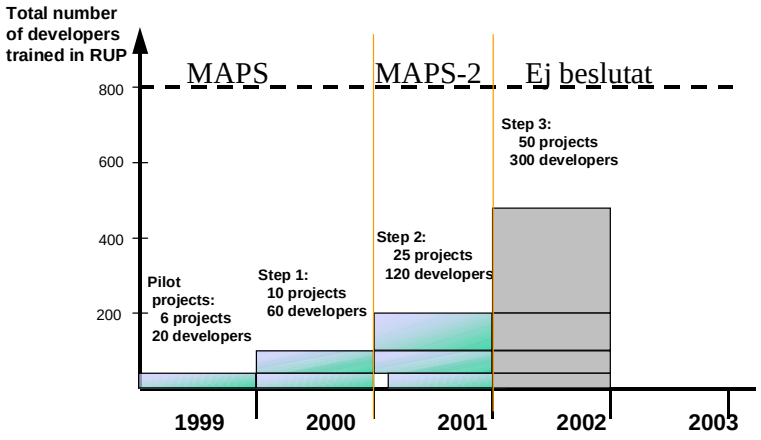

Figure 12 RUP implementation plan (from Volvo IT, 2002d)

RUP is implemented in the organisation in increments (steps) (Figure 12). The first pilot projects were used to evaluate the RUP method and make a decision of whether to implement the method in the whole organisation or not. The implementation of the third step has been slowed down due to problems with finding suitable RUP projects to start ("Ej beslutat" at the third step in the figure means "not decided").

One of the keys of implementing the method is to always have developers that are experienced with the method participating in each project. The project members that gain experience in using RUP, especially those that become "RUP coaches", participate in new projects and introduce the method to new developers. The MAPS project (described below) provides the resources and support needed. The competence in the organisation is in this way build up incrementally.

The key to successfully implementing the RUP method has been a program of "Workshops and Reviews" offered to each project (Volvo IT, 2001b). The purpose of the workshops is to refresh the theory for each new major activity, and to help the project team in applying the theory on their own problem. In addition to the workshops and other coaching activities, the coaches and specialists also perform a set of reviews, aiming at assuring that the project builds the right product, and that the process and method is understood and applied in the right way.

The implementation of RUP in development projects is done with the help of the Methods And Process for Systems development (MAPS) project. This project maintains a network of "RUP coaches" and "RUP specialists" and provides method support and resources such as training, licences, tools, and coaching. The project was originally called MAPS but was then reorganised and then called MAPS-2.

The MAPS-2 project is focused on the implementation of RUP as a development method. Projects that implement RUP are supported during the whole life cycle of development (Figure 13) (Volvo IT, 2000). Before the project starts and in its first phase, developers that have not previously used RUP are trained in its use. Training in

the necessary tools is also provided. Workshops are conducted to refresh the theory just before it is needed, both as training in artificial settings and as user workshops where actual project work is conducted. The MAPS network furthermore provides skilled reviewers that review the RUP artefacts and milestones as needed.

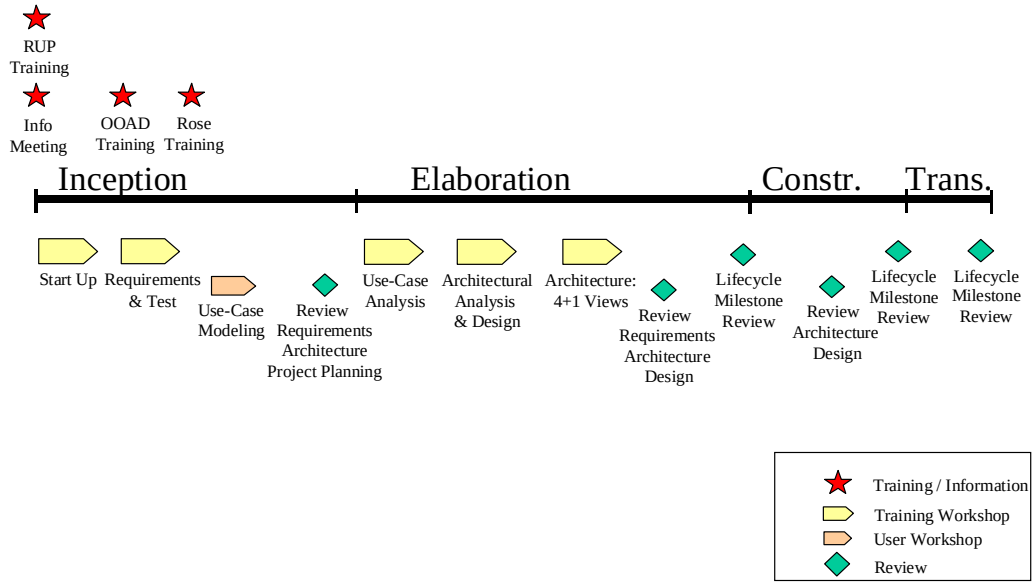

Figure 13 Support for the implementation of RUP (from Volvo IT, 2000)

The long-term purpose of a MAPS coach is to support the use of RUP in his/her work location, giving practical assistance to project members, and informing about new developments, plans, and results. The short-term objective is to learn how to coach and becoming skilled in the RUP process by participating in RUP projects, and by exchanging experiences with other coaches in the MAPS network.

A coach has to have an understanding of the results that have to be produced in each workflow, know the level of ambition appropriate in each phase and iteration, and see which tools are appropriate depending on the character of the RUP project and the project member's prior experiences. The coach has to be experienced with systems development and understand and know the RUP method. Furthermore good personal qualities are important, such as leadership abilities (to make things happen and focus on the results), communication skills (to be able to listen, win confidence, motivate, and convince), and educational skills (to be able to put theory into practice).

The work as a RUP coach can be divided into different coach roles as follows:

- Project support coach
    - Plans, together with the project manager, workshops and reviews to be done in the project
    - Should be able to solve most of the problems, make most of the decisions and answer most of the questions that appear in the project (except in the first project when learning the method)
    - Is usually supported by another RUP specialist (RUP consult or specialist coach)
    - Keeps a log of coach activities, agreements, problems, and solutions to problems that have occurred
    - Gives continuous support to the development projects regarding the RUP method and other methods and tools used in the project
    - Is responsible for weekly updates of the project together with the RUP specialist
    - Has responsibility for collecting questions that appear in the project and find solutions to them using the recourses of the MAPS network
- Role support coach
    - Gives workshops on one or several roles/workflows in RUP
    - Can support projects in one or several roles/workflows
    - Carries out specific planned activities in different projects
- Specialist coach (Process Engineer)
    - Primarily takes care of exceptional situations that can arise in a project
    - Promotes good competence development, coaching other coaches
    - Works part time in a project with specialist efforts
    - Understands how the RUP method should be tailored for different types of projects
- Tools coach
    - Needs to be familiar with the functionality of the tools used and know how they interact
    - Knows how to set-up and help new co-workers with project specific templates etc.
    - Is a good teacher
    - Knows how to use the RUP Project Kit (RPK)
    - Knows how the tools are used in the development environment

Volvo IT had initially planned to introduce RUP to the whole organisation in the year 2003, but that will not be possible. This is mainly because there have been fewer projects than expected that are suitable for implementing RUP. The focus for the third step of the implementation plan (Figure 12) has been on maintaining the current level of competence. Ensuring that those developers that have participated in RUP projects

keep their competence by participating in new RUP projects does this. Furthermore the introduction of RUP to new developers continues albeit slower than originally planned due to the problem of finding suitable projects. Eventually the RUP method and the PCM method will be used by the whole organisation in all locations for all application development.

6.2.5 How RUP is adapted and used at the organisational level

The Common Skills and Methods department in Göteborg is responsible for providing methods for use in Volvo IT. This means that they take care of providing the method descriptions and tools (and the relevant licences for commercial ones). The methods are either developed in-house or bought from commercial vendors.

While it is the Common Skills and Methods department that is responsible for this, the work can be distributed to other departments in Volvo IT, in order to make better use of resources and competence. For example, the adaptation of RUP for each development environment and platform is done mostly at the location where it is most needed and the best competence exists. The MAPS network is also distributed in all locations of Volvo IT.

The Common Skills and Methods department can be said to provide a modification of RUP at a global organisational level. The department provides the guidelines for implementing and adapting the method. These are made accessible to all Volvo IT departments through the Internet. This information is only accessible for Volvo IT employees and consultants.

Volvo IT has made a mapping of the RUP roles to 9 more abstract roles. This mapping is considered more relevant for the type and size of the projects conducted at Volvo IT and is recommended for use in all projects. This mapping is shown in appendix C (adapted from Volvo IT's method support web site). In this mapping, only 17 of the 33 roles currently prescribed by RUP are included. The mapping does not include any of the reviewer roles, because reviews are mostly done in the organisation in other ways than those described by RUP. Project artifacts are rather reviewed by external reviewers provided by the MAPS network.

The RUP method is used in conjunction with the PCM method used for project administration and management. A mapping has been made which explains how to use the methods together and which parts of RUP are replaced with PCM.

The organisation does not do any major modification to the RUP product itself or the document templates. The only adaptations made by the organisation to the document templates are to the title page and the heading of the pages, while leaving the structure of the content intact. The reason for this is because the RUP method is constantly evolving and Volvo IT considers it to be too much work to modify the method in detail since this modification would have to be updated for new versions of the RUP. Furthermore, major modifications would make it harder to hire consultants experienced with RUP or to cooperate with other organisations using RUP (since they would have to familiarize with the modifications).

6.2.6 How RUP is adapted and used at the divisional level

Each division (each location) of Volvo IT is free to make some minor adjustments to the RUP method to make it better fit onto the specifics of that organisation. For example at the Skövde division there exist a number of documented work procedures that are somewhat specific and traditional for that work place. These will be kept in

use since RUP does not support the same procedures or is not sufficiently specific for the situation at Volvo IT in Skövde.

As mentioned before, the adaptation of RUP to different development environments is mostly carried out at the divisions and departments that use those most and/or have the best competence. Many of these adaptations are then specified in guidelines that are made available to the other divisions (at the organisational level). The other divisions are required to follow the guidelines, but may add modifications of their own. The result is that divisions of Volvo IT use RUP basically in the same way allowing small differences to better suit each workplace.

6.2.7 How RUP is adapted and used at the project level

All RUP projects are staffed in such a way that there is always a participant that has experience of using RUP and can instruct other participants. This is done with the help of the MAPS project that also provides courses, workshops, and reviews to support the project in method usage (as discussed earlier).

The project plan is always done according to the PCM method, which is used for project administration and management, so most of the project management parts of RUP have been replaced by PCM. Some other parts of RUP may also be replaced with parts from the PCM method if that is considered fitting in specific projects. For example can projects use risk management according to PCM instead of RUP.

At the start of each project there is a start-up workshop where the project team decides what parts of the RUP method will be used in the project. More specifically, they decide what roles, activities, artifacts, and tools are needed. All these decisions and modifications are summarized in a RUP document titled Development Case. Other things are also considered, which are not a part of the RUP method (e.g. installation of cables, devices, etc.), and planned for as well. Those are documented in the Project Charter, which is the central project plan for each project according to the PCM method.

The PCM method only concerns the project manager; other developers follow the prescriptions of RUP for their roles. Since it is decided in the beginning what to do, there is not need of any major modifications of the method later in the project, and developers can proceed according to the Development Case and the Project Charter.

6.2.8 Examples of the adaptation of RUP at the project level

This example is taken from a project that is being conducted at Volvo IT in Skövde. The project is, at the time of writing, in the Transition phase (nearing completion). The Development Case document for the project describes how RUP is to be employed in the project (Volvo IT, 2001a).

The Development Case provides a description of how the core workflows are to be employed in the project. For this particular project, all the core workflows prescribed by RUP are used but with some modifications. The biggest modifications were that Business Modelling was only to be used to model the domain and that Project Management was adopted to support the PCM method. Other workflows have only minor modifications.

For most of the workflows some activities have been removed. Crossing them out on a figure showing the activities of the workflow indicates this. Figure 14 shows an example of this in the case of the Project Management workflow. The workflows have been removed because they are not considered justifiable for this particular project or

because they have been replaced by workflows of other methods than RUP (PCM). For example, Project Approval Review and Lifecycle Milestone Review were not necessary because similar activities are done according to the PCM method.

Figure 14 Examples of workflows removed from RUP (from Volvo IT, 2001a)

The artifacts to be used in the project are listed in tables for each core workflow. An example of this is shown in Table 3. The table shows which artifacts to use for the Project Management workflow, level of use in each development phase, formality of the document, and which tools are used.

Table 3 Example of artifacts to be used in a project (from Volvo IT, 2001a)

Artifacts	How to Use				Review Details	Tools Used
	Incep	Elab	Const	Trans		
Business Case	Could	Could	Won't	Won't		
Iteration Assessment	Must	Won't	Won't	Won't	Informal	Word
Iteration Plan	Must	Must	Must	Must	Formal-Internal	Word
Product Acceptance Plan	Could	Could	Won't	Won't	Formal-External	Word
Project Measurements	Must	Must	Must	Must	Formal-Internal	PCM Monitoring
Quality Assurance Plan	Could	Won't	Won't	Won't		
Review Record	Must	Must	Must	Must	Formal-Internal	Word
Risk List	Must	Must	Must	Must	Formal-Internal	Requisite Pro
Risk Management Plan (PCM)	Must	Must	Must	Must	Formal-Internal	Word
Status Assessment	Could	Won't	Won't	Won't		
Project Charter (PCM).	Must	Must	Must	Must	Formal-Internal	Word
Business Agreement	Won't	Won't	Won't	Won't	Formal-Internal	Word
Steering Committee Plan	Must	Must	Must	Must	Formal-Internal	Word
Name Standard	Must	Must	Must	Must	Formal-Internal	Word
Result Report	Won't	Must	Must	Must	Formal-Internal	Word
Final Report	Could	Could	Could	Must	Formal-Internal	Word

An interesting thing to note in this example is that some additional artifacts have been added to the ones prescribed by RUP. These are Project Measurements, Risk Management Plan, Project Charter, Steering Committee Plan, and Name Standard (marked with italics in the table). These are all documents that are used according to the PCM model.

The RUP artifacts *not* to be used are listed in other tables. Table 4 shows an example of this for the Project Management workflow. The table shows the artifacts to be excluded from the development process and the reason for excluding them. The table shows the RUP artifacts that have not been considered relevant or have been replaced by artifacts from the PCM method or something else.

Table 4 Example of artifacts *not* to be used in a project (from Volvo IT, 2001a)

Artifacts	How to Use	Reason
Measurement Plan	Won't	Ideally there should be a measurement plan on organisation level to ensure that all projects are collecting the same metrics. This does not exist and therefore we don't expect the projects to produce a measurement plan either but we want them to do measurements.
Problem Resolution Plan	Won't	This should be covered in the project charter.
Work Order	Won't	This level of ceremony is not normally used at Volvo IT.
Software Development Plan	Won't	Replaced by PCM Project Charter
Iteration Assessment	Won't	Replaced with PCM result report, and after each iteration a new version of result report will be produced.
Quality Assurance Plan	Won't	Quality Assurance Plan Replaced with Project Charter/Principles and plans affecting the project quality
Status Assessment	Won't	Replaced with Steering committee meeting
Business Agreement	Won't	Removed from the project, and replaced with an order and business requirement specification.

The Development case also describes which roles are to be used in the project. At the time this project started, RUP defined 31 roles. To avoid having to assign staff to all these roles a mapping has been made that defines a set of higher level of roles that were used for staffing. This mapping is shown in table 5, which shows the name of the mapped role and the names of the RUP roles it represents.

Table 5 Example of mapping of roles (from Volvo IT, 2001a)

Roles on VIT	RUP roles
Analyst	System Analyst, Use-Case Specifier & Tester
Developer	User-Interface Designer, Designer
Project Manager	Project Manager & Deployment Manager Change Control Manager
Architect	Architect
Implementer	Implementer, Tester
Test Designer	Test Designer
Process Engineer	Process Engineer
Configuration Manager	Configuration Manager & System Integrator
Toolsmith	Toolsmith

It is interesting to note that this mapping only includes 16 of the 31 roles prescribed by RUP (the version of RUP used by this project) and results in 9 combined roles to be used. This mapping is slightly different from the mapping prescribed by the

organisation at the time of this writing. The difference is that Change Control Manager is part of the Project Manager role instead of the Configuration Manger role and that Capsule Designer is not a part of the Developer role as in the organisational mapping.

The document separately lists the Reviewers needed in the project, in this case Requirements Reviewer, Design Reviewer, Architecture Reviewer, and Project Reviewer. This accounts for four more roles used in the project.

The Development Case also lists some of the roles that are not used and the motivation for that decision. For example the roles of Business Process Analyst and Business Model Reviewer have been excluded since Business Modelling is done using the CAMP method instead of RUP. Other roles are not used because they have not been considered applicable for this project.

6.2.9 How RUP is adapted and used at the individual developers level

Since the prescriptions of activities and document templates in RUP are not modified by the organisation, it is evident that individual developers have to use them as they are or modify them themselves. By viewing some samples of the project documentation it seems that developers do not do everything as prescribed by the method. Some interesting modifications, anomalies, or observations will be mentioned below. It is important to note that many of the documents are from a project that has not been completed (but almost). Thus, some of the documents can not be considered as complete.

The overall structure of the RUP documents that were reviewed has not been modified significantly. In some cases headings have been left empty. In some cases headings have been removed, and often sub-headings have been added under the relevant heading.

In some cases other documentation is referenced instead of filling in information under headings in RUP documents. These are often references to information in documents created using the PCM method.

It seems that the document templates are often unnecessarily large for the size of the project. An example of this is found in the Vision document for one of the projects. The document has a title page, page for revision history, and table of contents. In the document there are eleven level one headings each with up to eight level two headings (sub-headings). Very often the text under the headings in the documents consists of only one line of text or a reference to another document. In other cases it may be left empty. This is also quite common in other RUP documents. The information has been filled in under the relevant headings of which none have been removed from the document template, so the structure of the document is intact. Five level two headings have been added and one level three heading. By stripping all headings and extra pages (title page etc.) and summarising the actual content of the document it could be shrunk from fourteen pages to just two pages of text. This shows that while templates can provide documentation aid, the resulting documents can be unnecessarily large. Shorter documents could be more appropriate and easier to read in these cases.

The revision history, which is a part of RUP documents, was not used in many the documents viewed. Instead a reference was made to the document management and version control tool used (i.e. MS SourceSafe). In some cases the revision history was used.

Some of the RUP documents are written in English, but the majority of the documents are written in Swedish. The headings of the documents have on the other hand not been translated to Swedish.

Some of the project documentation created is not prescribed by RUP. An example of this is the User Documentation for the system being developed. These documents are in some cases written in Swedish and do not follow the format of RUP templates. There is also documentation in relation with to the PCM method, which follows the prescriptions of that method instead.

The motivation for the changes made to the method is usually not provided in the documentation. Such information will be dealt with in more detail in the interviews.

6.3 Results from interviews

The results from the interviews are presented in a number of categories listed below.

6.3.1 Background of the interviewees

All the respondents have worked as systems developers (sw. systemutvecklare) at Volvo IT for various times (4-13 years). Some of the respondents have, in recent years, worked with method support in the organisation and/or technique support (support the use of different techniques).

When asked about their experience of development methods, all the respondents except one have worked with the AU-model (see section 6.2.1), which as stated earlier is the in-house method of the organisation. One of the respondents had tried to work with a method called iAD, which is an iterative model that makes use of three-month time boxes and checkpoints. Another respondent had not used the AU-model, since that person had used RUP from start at Volvo IT. The respondents had not used any other development methods, besides RUP.

The respondents have varying experiences of using RUP. All the respondents have worked on one or more of the four first RUP projects at Volvo IT in Skövde. These projects have been conducted by between 4 and 12 developers in the years 1999 to 2002. The project time for the three first was between 6 and 18 months and the fourth project has run for 3 months but is temporarily paused.

Two of the respondents have participated in three RUP projects, including the first one, which was a pilot project. These two persons have also worked as a coach in their later projects, i.e. educating other project members and providing support in the use of RUP while participating in the project. The other two respondents have participated in one and two projects respectively.

6.3.2 Use and modification of roles

The respondents described that the 34 roles prescribed by RUP are mapped to 8 more abstract roles [The current organisational mapping actually includes 9 roles][8]. This is done in order to simplify things and to avoid puzzlement over 34 different roles [The current RUP version includes 33 roles]. It was considered to be conceptually easier to have fewer roles. The roles used are: Project Manager, Process Engineer, Analyst, Developer, Implementer, Architect, Test Designer, and Configuration Manager. The smaller related roles are then mapped to these roles.

[8] Brackets indicate comments made by the author that add information to what the respondents said.

This mapping was described by one of the respondents to be due to the fact that the projects at Volvo IT are usually small, and therefore better suited for 8 roles. Each of the 8 roles summarise the RUP roles that are logically connected and are therefore usually suitable for one person to perform. This is similar to the way they have worked according to the AU-model before. Developers that worked with the AU-model had certain responsibilities and have kept those responsibilities even when working with the RUP model (with the related role). So the mapping of the "sub" roles to "super" roles reflects the previous work procedures at Volvo IT. This mapping of the roles is documented on the organisations web site. The documentation lists the 8 roles and their "sub" roles, provides a description of the roles and summarises which artifacts each role is responsible for.

All RUP projects at Volvo IT use this set of 8 roles. There are no other major modifications to the roles from the RUP model. To compensate for different number of developers in projects, persons either have many roles or share roles in between them. Usually there is one role per person, but often persons share roles, e.g. two Analysts and two Implementers. When roles are shared, the developers sharing the role usually become responsible for different artifacts for that role, rather than creating all artifacts together.

One respondent stated that the boundary between roles is not always clear. For example the boundary between Designer and Implementer is not clear, since the design was expressed in a graphical tool and the source code is then generated from the tool.

One of the respondents said that they may have misunderstood the purpose of the roles in RUP. They have considered a role as being at the same level as a profession. Therefore they considered it best to have just one role, and play that role in all projects. This person later considered the roles more in terms of a set of instructions for different situations in the project, rather than a profession.

One respondent stated that developers in Sweden are not used to working with roles. The developers in a project, except the project manager, are simply systems developers (sw. systemutvecklare) rather than having different roles. In this sense identifying 8 roles is a big step. This enables developers to specialize in an area and makes it unnecessary for each developer to know the whole development process in detail.

6.3.3 Selection, use, and modification of activities

Developers in the RUP projects do not perform all the activities prescribed by RUP. One of the respondents described, in some detail, how the activities to perform are chosen. At the start of each project there is a "start-up" in which the Process Engineer and possibly the Project Manager (if it is not the same person) select which activities and artifacts will be needed. The expected result of the project has the biggest effect on this selection because everything selected has to generate something useful for the development of the system or related artifacts. The whole project group then goes through the selection so that everyone has the same understanding of what is to be done. These selections of roles, activities, and artifacts and their motivations are described in a RUP artifact (document) called the Development Case. This document also describes additional activities and documentation that is needed, which is not part of RUP. For example a document called Project Charter from an internal project management model called PCM always replaces the Software Development Plan document in RUP.

One of the respondents stated that in their first RUP project, they tried to do everything as prescribed by the method. But then they noticed that they spent much time on producing documents and realised that they also had to produce a software system. After that they have been more careful to consider that everything they do in the project must bring some value to the system they are producing. This makes the selection of what to use from RUP and what to skip an important aspect of using RUP.

When the selection of what to do has been made at the beginning of the project, each developer can usually just follow that selection without changes. In some cases the selection can be changed during the project because the developers are learning and experimenting with the RUP method.

Respondents were asked about whether the prescriptions in RUP provide the necessary information about how to perform activities for a role. They said that it was often the case but not always. RUP describes a lot of *what* needs to be done, but often it is not described in detail *how* to do it. It was also mentioned that sometimes there was so much to choose from that it was difficult to see what to use for each situation.

Two of the respondents said that they have had to add activities that are not prescribed by RUP. This was often activities needed for the installation of the system, e.g. preparation of installation plan and instructions and installation of cables, computers, and operating systems. These things had to be in the plans, even if other departments of the company were responsible for them, but were not supported by RUP. One of the respondents said that RUP lacked in support of requirement management and the descriptions were minimal concerning the design and later activities, because those activities are dependent on the implementation platform. When complementing RUP, earlier conventions and methods of the organisations were used.

Respondents were asked if they perform the activities prescribed by RUP as they are prescribed. The answers indicate that they try to do so when possible, but usually they do things somewhat differently. One of the respondents said that it was not possible to do everything as prescribed because their situation and organisation did not need it. That respondent perceived RUP as describing how to build a gigantic system, like something for NASA, while Volvo IT usually builds rather small systems. There may therefore be irrelevant parts that must be removed from the method, up to 80% in some situations (according to one of the respondents), and this makes it hard to work with RUP.

In some situations the activities must be changed to a large extent. In those cases the developers use their old ways of doing things. One of the respondents said that RUP is very general in order to be useable everywhere, which makes it necessary to adapt the method to the specific circumstances of the organisation as well as to each project. Some activities are not supported by RUP, e.g. defect tracking, so developers have to complement RUP with other methods when that is needed.

Another respondent said that one builds on ones earlier experience when doing the activities prescribed by RUP. You look into RUP to follow the steps to do the activity. You do not follow the steps exactly, but rather use RUP as a checklist, either in parallel or afterwards. This respondent also said that they usually do not make major changes to the important activities they have chosen. Rather they try to follow activities as prescribed.

6.3.4 Use and modification of document templates

The respondents were asked to describe how they use the RUP templates. This can be summarized as follows. They read through the template and use the headings in the document mostly as they are and fill in the information they know. Other necessary information is collected later and filled in. Headings that are considered unnecessary for the particular project are removed. Others are left empty if it is possible they will be needed later in the project. Examples provided by RUP or documents from similar projects at Volvo IT also give hints about what can be excluded or needs to be added. Sub headers are created as needed for additional information. Since there can be some changes in the project and since the knowledge of the system increases with time, they go through the document in each phase of development to see if anything needs to be added, taken away, or changed. The structure of the documents is not modified (except title page and heading), because the templates are used as a standard within Volvo IT.

One respondent said that it was often hard to know how much to work on a document. In RUP there are numbers that state in, percent, how much of a document should be ready. But what does it mean to have a document 50% ready? Are 50% of all the headings filled in or all headings filled in to half? Such issues could start philosophical discussions during the projects.

6.3.5 Relation between RUP and other methods at Volvo IT

Respondents were asked how RUP was related to other development methods at Volvo IT. One respondent described the relation between methods at Volvo IT in some detail. Volvo IT has made a mapping between RUP and PCM, an in-house method that is the master administration method at Volvo IT [see chapter 6.2.3]. This has also been done against other methods, for example MCM that is their maintenance method, and CAMP that is a method for conducting different types of pre-study, investigations, and modelling. The results from the CAMP method may be input to RUP if a decision is made to create a software system.

Some respondents also mentioned the AU method, an earlier development method used at Volvo IT [see chapter 6.2.3], and which most of them have used. One respondent said it was easy to fall into old habits and do things according to the old method.

One respondent mentioned that they have in-house methods for the design specific to the development environment used. For the Windows DNA environment they have specific descriptions/guidelines of how to go from the design to code. The Common Skills and Methods department in Göteborg is responsible for this material.

6.3.6 Use and modification of RUP in general terms

Respondents had little to add when asked about how they use and adapt RUP in more general terms. To summarise they extract from the method that which is of use in order to produce the software system and remove everything of less use. A major part of the decisions about the usage and adaptation of RUP is described in the Development Case document and Project Charter, which the Process Engineer and Project Manager are responsible for. Other developers usually do not have to make these decisions, but rather follow the Development Case, Project Charter, and the RUP prescriptions for their roles. The projects receive support for method usage in form of courses, workshops, coaches (experienced with RUP) working in the project

providing support on site, and external reviewers (usually coaches) that review the RUP artifacts.

Some respondents said that when using RUP it works a lot like a checklist. You think through what needs to be done and then read through RUP to ensure you have not missed anything.

6.3.7 Factors affecting the use and modification of RUP

Factors affecting the selection of activities

One respondent said that the size of the project and in some cases the time available had the greatest effect on the selection of activities.

One respondent said that the feeling that some other person would use the artifacts produced had the greatest effect on how and why activities are chosen to be done. If a person has a feeling that the artifact produced will not be useful, that person will rather choose to do something else that is considered important.

Another respondent said that personal knowledge of RUP had the greatest effect on the selection of activities. They often had to try different things or improvise in the beginning of using RUP. They sometimes had to abandon ongoing activities after realising that it was a waste of time. This is the reason why it is important to have a good coach or guide in a project that knows which activities should be selected and which should be skipped. The same person also said that developers' earlier experience have great effect on how activities are selected. Sometimes there may even be conflicts when experience says that it is better to back up and do some things over again but RUP says to move forward.

Factors that affect how activities are modified

Respondents were asked which factors have the greatest effect on how activities are modified. One of the respondents said that they make a judgement of what is relevant for the type of system being developed. Furthermore, they do not think in terms of modifying the method, they rather think of how they have done things in the past.

One of the respondents said that the usefulness for the final result and the size of the project have the greatest effect. If you are developing a new and large system, it is not well known what can be excluded from the method. This means that many activities are included at the beginning of the project. Later in each phase or iteration of the project it may be easier to see which activities can be excluded. The more you build, the more you know, and the more you can exclude. So activities are excluded during the whole project time. The larger the project, the less you dare exclude, the smaller and delimited the project, the more you dare exclude from start.

One respondent said that you must build on earlier experience. In the fist RUP project that you participate in it is difficult to use and understand RUP. It takes two to three projects to understand RUP and its usefulness.

Factors that affect how document templates are modified

The respondents were asked which factors have the greatest effect on how templates are used and modified. Factors mentioned were the purpose of the document (what should be in the document), the Development Case (which describes how RUP should be used in the project), the nature of the project (only information relevant for the project is put into the documents), and earlier experience (if they have written documentation before).

The templates are used as a standard at Volvo IT, and therefore the structure of the documents are kept intact and only small modifications made. The only modification made by the organisation is the title page and the header of the pages. The organisation considers it important that the templates are changed as little as possible in order to make it easier to share resources between divisions. In this way everyone knows what the artifacts stand for and what to be expected of the role responsible for the artifact.

Factors that affect how RUP is adapted in the organisation

One respondent said that the method support departments [i.e. Common Skills and Methods in Göteborg] were the biggest factor affecting the adoption of RUP. That is due to the fact that they have made most of the preparation work of how RUP is to be used at Volvo IT. Persons [coaches] from these departments work in the projects and support them. The same interviewee also said that Volvo IT is a large organisation and there will always be some islands where people do things their own way to some extent. So even if the bigger directives are managed centrally and they cooperate a lot, development will never be done in 100% the same way.

One respondent also said that RUP is mostly adapted to how people have worked before. The administration and persons responsible for method support at each location have big effect on how RUP is used.

Other factors mentioned were the usefulness of RUP with regard to the final result and the size of the project.

Problems with the use of RUP

Some problems of using RUP were mentioned in the interview discussions. Some of these are discussed below.

Respondents considered it hard to start using RUP. One respondent stated “It is hard to see the red thread in RUP in the beginning [of using the method]” (the author’s translation). One respondent said that after participating in a project for about 1500 hours [about 10 months] a developer has gained good understanding of how RUP works, how activities are connected and the general flow of development. Others also considered one sufficiently large project enough to gain basic understanding of RUP but the respondents considered participation in 2-4 projects necessary before being really skilled in using RUP.

One respondent mentioned that they had problems with the large number of roles that are in RUP. This is the basic reason why Volvo IT has made a mapping of the roles more suitable to the organisation.

One respondent said that the biggest problem with using RUP is that you can drown in all the method descriptions. If most of the developers are new to the method a lot of the time goes into ideological discussions of how to do things. Many descriptions of RUP can be interpreted in different ways and there is often a large gap between *what* needs to be done and *how* to do it.

Respondents mentioned that there is often lack of support of *how* to do things in RUP. This is especially problematic if a person is new to the method. One respondent actually said that the idea with RUP is not to describe *how* in all situations, but rather that you have to figure that out yourself. When using RUP for the second time, you consider it differently from the first time and it is possible to form your own way of working with the method.

One respondent mentioned that they had problems with the iterative way of working, since they had not worked iteratively before. Therefore, it was sometimes difficult to see how deep it was necessary to go in activities, e.g. analysis, before continuing in the next iteration.

One respondent said that it is important not to forget earlier experience and to use common sense when using RUP. Another respondent said that you must not be frightened by all the documentation and activities in RUP but concentrate on the parts that provide value for the development of the product.

Some other specific problems mentioned were that it is difficult to: Maintain the traceability between Use Cases, Analysis Model, and Design; start using Use Case Realisation; understand the 4+1 views; understand how the design should be implemented; and to know which information should be included in the documentation.

7 Analysis of results

In this chapter, an analysis of the case study results is presented. The results are put in context by comparing them with results from related research, and by comparing them to some recommendations in the Rational Unified Process®[9].

7.1 Results in context of other research

The results of this project are in many ways similar to the results of other research.

Other research has shown that development methods are usually not used exactly as prescribed (e.g. Bansler & Bødker, 1993; Fitzgerald, 1998; Power & Richardson, 1996; Russo et al., 1996; Westrup, 1993). This project's results also show that methods are modified to fit the specifics of the organisation and each development project. This project adds one example of this described in some detail.

When compared to the Motorola case described in chapter 2.4.4 there are some interesting similarities. In both cases the adaptation of the methods used can be described as being done at different levels. The adaptation of the methods is done first at a *global organisational level*. In the Motorola case this is termed the Industry level where methods are taken from the public domain, known software standards and models (e.g. IEEE 1074, CMM, and V-SLCM), and adopted into the organisations method (OSSP). This corresponds to the Organisational level at Volvo IT, in which they adapt RUP and a number of in-house methods for use in the organisation. The next level is a *local divisional level* to take into account the specifics of different divisions (sites) in the organisation. The Motorola case terms this the Organisational level, where a number of software processes specific to the various divisions within the organisation are added to the organisations method. This corresponds to the divisional level at Volvo IT where work procedures and adaptations of methods are in place but can become a part of the organisational level (usually as guidelines for using RUP with different development platforms). In both cases the project management is responsible for tailoring at the *project leve*l by choosing the necessary parts of the methods and making micro level modifications as needed.

The Motorola research did not identify and discuss the roles of the individual developers in the projects. In this aspect, this project has gone one step further and identified a fourth level of adaptation, where individual developers make their own small modifications to the method and use the method in somewhat their own way.

The results of this project fit well into the framework for systems development described in chapter 2.4.5 (Fitzgerald, 1998, 2001). The basis for development at Volvo IT is based on methods, both commercial ones (RUP) and in-house ones (e.g. PCM, MCM, CAMP). Even if the base is the same no two projects are exactly the same, there is rather a unique method-in-action enacted by the developers in each project. The specific development context shapes the method-in-action, especially due to the focus on the final result and the usefulness of the methods. The methods and the roles prescribed also influence how development is enacted. The results of this project do not contradict this framework for systems development, rather provides an example that supports it.

[9] Rational, Rational Unified Process, and RUP are trademarks or registered trademarks of Rational Software Corporation in the United States and/or other countries.

The results of this project can be considered to fit, to some extent, into the terminology of method engineering (discussed in chapter 2.4.3). Method engineering proposes solutions for development by engineering situational methods tuned to the situation at hand by assembling the development method with standardized building blocks and guidelines, so-called method fragments. In the Volvo IT case, parts of in-house methods and RUP were constructed differently for each project. The results therefore provide an example of how this is done in practice. The difference is that Volvo IT bases this construction mostly on experience and common sense rather than following a framework, tool, or mathematical languages, which some researchers of method engineering have proposed (e.g. Brinkkemper, 1996; Brinkkemper et al., 1999; Ralyté & Rolland, 2001).

7.2 Results in context of recommendations in RUP

The newest version of RUP (2002.05.00) includes recommendations on and support for how to implement, adapt, and use the method. This is mostly covered by descriptions of how to implement RUP, both at the organisational and project level. By comparing some of the recommendations in RUP to the project results it is evident that Volvo IT has followed many of the recommendations. Some of these will be discussed below.

RUP states that implementing a software development process in an organisation is a complex task and needs to be done in a controlled manner (Rational, 2002). RUP recommends treating the implementation as a project that is external to or is a sub-project of software development projects. This is what Volvo IT has done, by setting up the MAPS project and making implementation plans for RUP in different phases.

Furthermore, RUP recommends a number of different approaches of implementing the method in an organisation, one of which is to run more than one pilot project before real projects start using the new process and tools. This has been done by Volvo IT, using the results of six pilot projects to evaluate the impact of implementing the method in the organisation before starting the real implementation. Another approach described by RUP is to develop and maintain a development environment for the entire organisation, and having a team to develop and maintain this organisation-wide environment consisting of processes, tools, and infrastructure. This is what Volvo IT has done through the Common Skills and Method department.

An interesting aspect of the implementation of RUP in the organisation studied is that it is a process spanning many years. The fact that it takes such long time is somewhat surprising, but when one takes onto account the risks and effort involved, it seems reasonable being careful.

At the project level, the Development Case is central in the adaptation of RUP. The purpose of the Development Case is to capture the tailored process for the individual project and RUP provides guidelines and examples for how to do this. Volvo IT has apparently used these guidelines and the Development Case is central for the use of RUP in each project.

RUP includes many other possibilities for adaptation. It is, for example, possible to customize the document templates and even customize the product itself (the web based version). However, the organisation does not make any major modifications to the product itself due to the fact that it would require much rework for each new version of the product. Furthermore, it would limit the possibility to cooperate with other organisations using RUP if the methods were used differently.

8 Conclusions

8.1 Summary of results

This section provides a summary of the results of this study with regard to the aim and objectives.

Project aim:

> *To investigate how a widespread development method is implemented and used in an organisational setting*.

The aim of the study has been fulfilled by doing a case study in an organisational setting where a widespread development method is being implemented and used. The result is a case study description of how Volvo Information Technology implements, adapts, and uses the commercial development method Rational Unified Process®[10] in combination with other methods. The fulfilment of the objectives will in turn be discussed in more detail.

Project objective:

> *Investigate how a widespread development method is implemented in an organisation.*

The results of the document study provided the information to fulfil this objective. The major findings are listed below.

The organisation first conducted six pilot projects that made use of the RUP method (called RUP projects) instead of their own in-house methods. After the decision to use RUP as the central application development method in the organisation the implementation of RUP in the whole organisation started in an incremental way, building competence in steps rather than all in one go. This implementation started in 1999 and it will take some years to come before RUP has been taken into use in the entire organisation.

A central department in the organisation is responsible for supporting the implementation and use of methods (The Common Skills and Methods department in Göteborg). This department provides support through a project called Methods And Process for Systems development (MAPS). Projects that implement RUP are supported during the whole project. Before the project starts and in its first phase, developers that have not previously used RUP are trained in its use. Training in the necessary tools is also provided. Workshops are conducted to refresh the theory just before it is needed, both as training in artificial settings and as user workshops where actual project work is conducted.

The MAPS project also maintains a network of “coaches” and “specialists” that are skilled in the use of RUP and other complementing methods. These participate in their first projects to gain experience in using and adapting RUP and then participate in new projects to support them in various ways. The coaches can help setting up the project and make decisions of how to use and adapt RUP to the specifics of the project. Furthermore, they can participate in the project and provide support to other developers in their work by answering their questions and solving problems. The

[10] Rational, Rational Unified Process, and RUP are trademarks or registered trademarks of Rational Software Corporation in the United States and/or other countries.

coaches may also function as external reviewers that review the RUP artefacts and milestones needed for the RUP projects.

Currently, only projects considered suitable as RUP projects are implemented using RUP. Suitable projects are those that are sufficiently small, i.e. effort of 1000 – 3000 work hours, with 3-5 persons. Furthermore, 1-2 of the developers should be experienced in using RUP (preferably coaches) and 2-3 of the persons should be learning RUP during the project. Other projects make use of previous methods at Volvo IT. Eventually, as the competence in the use of RUP has been built up, all application development at Volvo IT is planned to use RUP.

Project objective:

> *Find out if the method is adapted to the organisation, and if so, which factors have effect on that adaptation*.

The results of the study showed clearly that the method is adapted to the organisation. The major findings of how the method is adapted and the affecting factors are discussed below.

The adaptation of RUP can be described as being done at four different levels: Global organisational level, local divisional level, project level, and individual developer's level (the latter two will be discussed in the next objective).

The Common Skills and Methods department administers the global organisational level. The RUP method is the central Application Development Process that provides both the framework and methods for Requirements & Analysis, Software Architecture, Design, Implementation, and Deployment. However, RUP is used in a combination with a number of in-house methods. The biggest modification of RUP is that an in-house method called the Project Control Model (PCM) is used on top of RUP for project management. PCM is seen by the organisation as providing more support for project management and reporting to higher-level administration than RUP does. Some of the activities and documentation prescribed by RUP for the Project Manager are therefore replaced by parts of that method. A mapping of which activities and documents are to be replaced between the methods and a description of how to use them together has been made by the organisation. Many other methods are in use at Volvo IT, which either support aspects of development that RUP does not cover or are considered by the organisation to provide better support for a certain aspect.

Another major adaptation is that the organisation has made is a mapping of the RUP roles to 9 more abstract roles. This mapping is considered more relevant for the type and size of the projects conducted at the organisation and is recommended for use in all projects. In this mapping, only 17 of the 33 roles currently prescribed by RUP are included, so some delimitation of the method has been done.

RUP is adapted to the development environment it is used in. This adaptation mostly results in guidelines on how to use the method in each particular development environment within the organisation.

At the divisional level, each division (location) of the organisation is free to make some minor adjustments to the RUP method to make it better fit the specifics needs of that division. For example at the Skövde division there exist a number of documented work procedures that are somewhat specific and traditional for that work place. These will be used continually since RUP does not support the same procedures or are not sufficiently specific for the situation at that division.

The central method support department of the organisation has great affect on how the method is adapted and used. That is due to the fact that they have made most of the preparation work and administer the method support in the organisation.

When adapting and using the method the organisation has a strong focus on the value the method brings for the development of software systems. Another major factor in this is how development has been conducted in the organisation before. The modification and use of the method in the organisation reflects this, e.g. by mapping the roles to be used in order to simplify them and simulating how the organisation has worked before.

Project objective:

> *Find out if the method is adapted to each development project, and if so, which factors have effect on that adaptation.*

The results of the study showed that the method is adapted to the specifics of each project. The major findings of how the method is adapted and the affecting factors are discussed below.

At the project level, RUP is adapted for the specific project at hand. At the start of each project there is a start-up workshop where the project team decides what parts of RUP will be used in the project. More specifically, they decide what roles, activities, artifacts, and tools are needed. Also they consider other things, which are not a part of the RUP method (e.g. installation of cables, devices, etc.), and plan for them as well. All these decisions and modifications are summarized in a RUP document titled Development Case and a PCM document called Project Charter, which is to be considered as the master project plan.

At the individual developers level the workers have to decide *how* to follow the RUP prescriptions of *what* should be done and even modify the details of following the method if needed.

The prescriptions in RUP do not always provide all the necessary information about how to perform the activities for a role. RUP describes a lot of *what* needs to be done, but often it is not described in detail *how* to do it. This leaves a lot left to the developers to figure out themselves.

The developers of the organisation try to use RUP as prescribed, but sometimes the activities prescribed have to be modified to fit the specifics of the organisation and the project. Other methods, work procedures, and conventions of the organisation are used to complement RUP as needed in the projects.

When using document templates in RUP, developers will typically fill in the parts that are relevant for the project and add other information as needed. Other parts are removed or left empty.

When adapting and using the method the organisation has a strong focus on the value the method brings for the development of software systems. A major factor affecting this is the usefulness of the method for each project, which in turn is dependent on the specifics of that project (e.g. project size, type of system, and available time).

Other factors affecting the adaptation and usage of the method in the organisation are how developers have conducted their work in the organisation before, their experience, and their knowledge of RUP. An experienced developer will know much better how to use and adapt the method while an inexperienced one will have to use more of trial and error.

Project objective:

> *Investigate how a widespread development method is used in an organisation.*

The study provides a description of how the method is adapted and used in the organisation, so this objective has been met. To summarise, the developers extract from the method that which is of use in order to produce the software system, skip parts of less use, and complement with in-house methods when the method is insufficient. A major part of the decisions about the usage and adaptation of RUP is done in the beginning of each project and described in the Development Case document and the Project Charter (project plan), for which the Process Engineer and Project Manager are responsible. The adaptation and recommendations made for the whole organisation has great effect on this in addition to the specifics of the project. Other developers usually do not have to make these decisions, but rather follow the Development Case, Project Charter, and the RUP prescriptions for their roles. The projects receive support for method usage in form of courses, workshops, coaches (experienced with RUP) working in the project providing support on site, and external reviewers that review the RUP artifacts.

Developers often use RUP a lot like a checklist. They go through what needs to be done and then read through RUP to ensure they have not missed anything.

8.2 Contributions

Since this is a singe case study, the results cannot be generalized to all other organisations, but in conjunction with other similar studies, the findings can be used to build theories on the subject. The findings therefore contribute to existing knowledge about how methods are implemented, adapted, and used in organisations. The results should therefore be of use both to researchers of methods and to practitioners that plan to use methods.

One of the strengths of this study lies in that a blend of techniques was used to collect data. In this, it enabled the researcher to collect more data and to identify if there were conflicts in that data. Strength lies in that the study was done in an organisational setting. This was considered to be very important for this study since the focus was on an actual organisational setting, while other types of research problems might call for other approaches. Another strong point is that the study provides an extensive picture of how a method is implemented, adapted, and used in a large organisation.

The results in whole are considered reliable by the author. The data was gathered from different sources, documents and interviews, instead of relying on only one source. This increased the reliability of the information since conflicting data could be identified and redundant data confirmed the information.

The document study made use of actual documentation of the organisation. While the documentation was of different age and providing different level of detail, no major contradictions were found that could undermine the results form the document study.

Only minor conflicts in data were found. These were partly due to the fact that the documentation gathered spanned four years, during which some changes have occured in the organisation. Furthermore, RUP was changed between versions. These conflicts could be resolved by validating the data with the informant in the organisation.

The number of documents and templates gathered was very large, about 200, and that was only a fraction of the documentation available from Volvo IT. A part of this documentation was selected for study and there is, naturally, some risk that important information was missed in this selection.

The interviews were recorded and then transcribed, which makes their findings more reliable than if only written notes and the interviewers memory were to be used. The author considers the findings to be sufficient for this project, even if many new questions came up during analyses of the interviews. Therefore, there is room to make a deeper investigation of method usage and adaptation.

The descriptions resulting from the case study were reviewed and validated by the informant in the organisation. The informant pointed out if information was misinterpreted, lacking, or incorrect. This ensured that the descriptions were correct (with higher probability) and thereby increased the reliability of the case study findings and their analysis.

A researcher's knowledge and experience will undoubtedly have some effect on the research conducted. Negative effects were minimised by keeping an open mind and by not assuming too much beforehand. It is considered by the researcher in this study, that prior experience with the RUP method has had positive effect on the results. That is because less time had to be spent on familiarizing with the method and consequently more time was available to concentrate on the actual study. Also because prior understanding helped the researcher to interpret the information gathered in the document study and to formulate questions for interviews.

The weakness of this study is that the scope of the project was very large and could therefore only be dealt with in rather general terms. A more specific investigation of fewer of the aspects of the topic could have revealed more details and deeper understanding. The document study and interviews only took place in one division of the organisation. If interviews had been conducted in more divisions, especially at the central department responsible for method support in the whole organisation, it is possible that more information would have been discovered and the study provided more details and understanding. The delimitation of only considering one division was made considering the resources available and the size of this project.

8.3 Possible future work

There is much room to investigate different aspects of the research topic and go deeper into those aspects. For example a continued study of the organisation could look deeper into how RUP is adapted and look into differences of how the method is employed at different locations of the organisation in different countries.

Doing similar research at other organisations could be very beneficial, enabling comparisons to be made of method usage in different organisations.

This project did not make an evaluation of the suitability of the RUP method itself. Most information about RUP comes from its vendor and the company's owners and can therefore be considered to be biased, since they are also selling RUP and related tools as products. Evaluating the method and comparing it to other methods or theories would provide important knowledge regarding the suitability and quality of RUP. Such an evaluation could be used to investigate whether RUP is the best choice for this organisation.

This project did not look specifically into how the organisation cooperates with the method vendors and how the method vendors support the organisation. This would be interesting to look further into, especially in order to further investigate if the organisation follows the recommendations of the method vendor and if the support is sufficient.

This project did not look into how CASE tools were employed in the organisation. Looking into how the methods and the tools are integrated and used would be interesting. This is especially because the method vendor also sells related tools and it would be interesting to make an independent evaluation of their suitability for use in the organisation and even compare them to other available tools on the market.

References

Avison, D.E., Wood-Harper, A.T., Vidgen, R.T. & Wood, J.R.G. (1998) A further exploration into information systems development: the evolution of Multiview2. *Information Technology & People*, 11(2), pp. 124-139.

Bansler, J.P. & Bødker, K. (1993) A reappraisal of structured analysis:design in an organisational context. *ACM Transactions on Information Systems*, 11(2), pp. 165-193.

Baskerville, R. & Stage, J. (2001) Accommodating emergent work practices: ethnographic choice of method fragments. In: N. Russo, B. Fitzgerald & J. DeGross (eds) *Realigning Research and Practice in Information Systems Development, the social and organizational perspective.* IFIP TC8/WG8.2 working conference on realigning research and practice in information systems development: the social and organizational perspective, July 27-29, 2001, Boise, Idaho, USA. Boston: Kluwer Academic Publishers, pp. 11-28.

Booch, G. (1999) Software development best practices. In: P. Kruchten *The rational unified process, an introduction*. Reading, Massachusetts: Addison Wesley Longman, pp. 3-16.

Bowen, J.P. & Hinchey, M.G. (1999) *High-integrity system specification and design*. London: Springer-Verlag.

Brinkkemper, S. (1996) Method engineering: engineering of information systems development methods and tools. *Information and Software Technology,* 38, pp. 275-280.

Brinkkemper, S., Saeki, M., & Harmsen, F. (1999) Meta-modelling based assembly techniques for situational method engineering. *Information Systems*, 24(3), pp. 209-228.

Brooks, F. (1999) No silver bullet: essence and accidents of software engineering. In: J.P. Bowen & M.G. Hinchey (eds) *High-integrity system specification and design*. London: Springer-Verlag, pp. 11-27.

Burman, R (2001) Experiences of CASE-technology and its use within Volvo IT. Lecture notes from guest lecture held at the Department of Computer Science at the University of Skövde, 8th November.

Fitzgerald, B. (1994) The systems development dilemma: whether to adopt formalised systems development methodologies or not?. In: W. Baets (ed) *Proceedings of Second European Conference on Information Systems*. Holland: Nijenrode University Press, pp. 691-706.

Fitzgerald, B. (1996) Formalized systems development methodologies: a critical perspective. *Information Systems Journal*, 6(1), pp. 3-23.

Fitzgerald, B. (1997) The use of systems development methodologies in practice: a field study. *The Information Systems Journal, 7(3),* pp. 201-212.

Fitzgerald, B. (1998) An empirical investigation into the adoption of systems development methodologies. *Information & Management*, 34, pp. 317-328.

Fitzgerald, B. (1998) An empirically-grounded framework for the IS development process. In: R. Hirschheim, M. Newman & J. deGross (eds) *Proceedings of the 19th International Conference in Information Systems.* December, Helsinki, pp. 103-114.

Fitzgerald, B. (2001) Method-in-Action: a framework for IS development. Lecture notes. Lecture held at Department of Computer Science, University of Skövde. 24th September.

Fitzgerald, B., Russo, N. & O'Kane, T. (2001) Software development method tailoring at Motorola. Forthcoming in *Communications of the ACM*. Available at Internet: http://afis.ucc.ie/bfitzgerald/cacm-method-tailoring-v3.doc [Accessed 2002-03-10].

Gummesson, E. (1988) *Qualitative methods in management research*. Sweden, Lund: Studentlitteratur.

Harel, D. (1992/1999) Biting the silver bullet: toward a brighter future for system development. In: J.P. Bowen & M.G. Hinchey (eds) *High-integrity system specification and design* (p. 29-51). London: Springer-Verlag.

Hares, J.S. (1994) *SSADM version 4: the advanced practitioner's guide*. England: John Wiley & Sons.

Iivari, J. & Huisman, M. (2001) The relationship between organisational culture and the deployment of systems development methodologies. In: K.R. Dittrich, A. Geppert & M.C. Norrie (eds) *Advanced information systems engineering, CAiSE 2001, LNCS 2068*. Berlin: Springer-Verlag, pp. 234-250.

Kruchten, P. (1999) *The rational unified process, an introduction*. Reading, Massachusetts: Addison Wesley Longman.

Patel, R & Davidson, B (1994) *Forskningsmetodikens grunder: att planera, genomföra och rapportera en undersökning* (2nd edition) [Foundation of research methods: to plan, accomplish, and report a research]. Sweden, Lund: Studentlitteratur.

Power, N. & Richardson, I. (1996) Methodologies are never enough. In N. Jayaratna and B. Fitzgerald (eds) *Lessons Learned from the Use of Methodologies*. Swindon: BCS Publications, pp. 247-256.

Pressman, R. S. (1997). *Software engineering: a practitioner's approach* (4th edition). New York: McGraw-Hill.

Ralyté, J. & Rolland, C. (2001) An assembly process model for method engineering. In: K.R. Dittrich, A. Geppert & M.C. Norrie (eds) *Advanced information systems engineering, CAiSE 2001, LNCS 2068*. Berlin: Springer-Verlag, pp. 267-283.

Random House (1984) *The random house college dictionary* (revised edition). New York: Random House.

Rational (2002) *Rational Unified Process* [on-line version] [version 2002.05.00] Rational Software Corporation. Available at internet: http://www.rational.com/ [Accessed 2002-04-08].

Rumbaugh, J., Blaha, M., Premerlani, W., Eddy, F., & Lorensen, W. (1991) *Object-oriented modeling and design*. New Jersey: Prentice-Hall.

Russo, N.L., Hightower, R., & Pearson, J.M. (1996) The failure of methodologies to meet the needs of current development environments. In N. Jayaratna & B. Fitzgerald (eds) *Lessons Learned from the Use of Methodologies*. Swindon: BCS Publications, pp. 17-22.

Sandström, C. (2001) *Systemutvecklarnas syn på rollindelningen i Rational Unified Process*. BSc dissertation no HS-IDA-EA-01-323. Skövde: University of Skövde. Available at Internet: http://www.ida.his.se/ida/htbin/exjobb/2001/HS-IDA-EA-01-323 [accessed 2001-02-22]

Sommerville, I. (1992) *Software Engineering* (4th ed.). USA: Addison-Wesley.

Tolvanen, J., Rossi, M. & Liu, H. (1997) Method engineering: current research directions and implications for future research. In: S. Brinkkemper, K. Lyytinen & R. J. Welke (eds) Method engineering: principles of method construction and tool support - Proceedings of the IFIP TC8 WG8.1/8.2 Working Conference on Method Engineering, Atlanta, USA, 26-28 August 1996, Chapman & Hall, London, pp. 296-317.

Volvo Data AB (1991) *AU-Modellen, projektering*. Volvo IT internal documentation.

Volvo Information Technology, North America (2002) [On-line] Available at Internet: http://www.sun.com/storage/success-stories/success-volvoit.html [Accessed 2002-04-03].

Volvo IT (2000) *Current software process improvement projects at Volvo IT*. PowerPoint presentation of method implementation, improvement, and support at Volvo IT, created by the Common Skills and Methods department 2000-03-15. Volvo IT internal documentation.

Volvo IT (2001a) *Development Case*. RUP Development Case document for a Volvo IT project. Created by Tjacobso 2001-12-10. Volvo IT internal documentation.

Volvo IT (2001b) *Assessing effects of RUP implementation using SPICE*. Created by Göran V. Grahn 2001-08-28. Volvo IT internal documentation.

Volvo IT (2002) *Volvo Information Technology*. PowerPoint presentation of Volvo IT, created by Ann Söderström 2002-02-05. Volvo IT internal documentation.

Volvo IT (2002b) *Metoder i Skövde*. PowerPoint presentation of which methods are used by Volvo IT, created by Ann-Britt Karlsson 2002-01-09. Volvo IT internal documentation.

Volvo IT (2002c) *MAPS 2001*. PowerPoint presentation of the MAPS-2 project. Volvo IT internal documentation.

Volvo IT (2002d) *Final Report*. Report about the results from implementing RUP at Volvo IT with the aid of the MAPS-2 project. Created by Gregor Börjesson 2002-02-12. Volvo IT internal documentation.

Westrup, C. (1993) Information systems methodologies in use. *Journal of Information Technology*, 8, pp. 267-275.

Wynekoop, J.L. & Russo, N.L. (1997) Studying system development methodologies: an examination of research methods. *Information Systems Journal*, 7, pp. 47-65.

References

Yin, R.K. (1994) *Case study research: design and methods.* USA: Sage Publications.

Appendix A – Interview questions (pilot)

Introduction

The interview will by conducted by Guðmundur Hallgrímsson. I am a student of Software Engineering (sv. Programvaruteknik) at Högskolan i Skövde.

This interview is a part of my final year undergraduate project (sv. Exjobb). My project is titled "The implementation, adaptation, and use of the Rational Unified Process at Volvo Information Technology - a case study". The use of the Rational Unified Process at Volvo IT has been selected as a case study.

This study has the following aim and objectives:

Project aim:

> *To investigate how a widespread development method is implemented and used in an organisational setting.*

Project objectives:

> *Investigate how a widespread development method is implemented in an organisation.*
>
> *Investigate how a widespread development method is used in an organisation.*
>
> *Find out if the method is adapted to the organisation, and if so, which factors have effect on that adaptation.*
>
> *Find out if the method is adapted to the specific circumstances in each development project, and if so, which factors have effect on that adaptation.*

This interview is mostly focused on the second and fourth objectives. Its purpose is to find out how each individual developer uses RUP and whether and how RUP is adapted.

Preliminaries

The interview findings will by treated confidentially. This means that your name will not be mentioned in the report in conjunction with what has been spoken. Therefore, I hope that you will be able to speak more freely when answering the interview questions.

The report will be written in English, so I would like to conduct the interview in English, but it is no hindrance to use Swedish.

In order to be able to concentrate on our discussions, instead of taking notes, I would like to record the interview with your permission. The recording will be used to transcribe and analyse the interview. The recording can be stopped at any time if you wish so.

My supervisor at Högskolan i Skövde is very interested in seeing the outcome of the interview. If you allow so, I will provide the summary of the interview to my supervisor without any link to your name. He will read it in order to better assist me in my project and possibly as input into his own research.

Appendix A – Interview questions (pilot)

The interview

First I will ask some opening questions to obtain background information. Then I will ask more specific questions, which refer to projects that you have participated in and which have been conducted according to the Rational Unified Process

Opening questions:

1. How long have you been working as at Volvo IT?
2. What have been your major tasks?
3. Which development methods have you worked with?

Specific questions:

4. How many projects using RUP have you participated in?
5. Which roles have you had?
6. What have been your major tasks?
7. Can the roles prescribed in RUP be used as they are?
    a. In what ways do you adapt roles in RUP?
    b. Why?
8. For the roles you have been assigned, do you perform all activities prescribed by RUP for that role?
    a. How do you select which activities to perform?
    b. What has the greatest effect on how these activities are selected?
9. Do the prescriptions in RUP provide sufficient information about how to perform the necessary activities assigned to a role?
    a. Do you add activities not prescribed by RUP?
    b. Where from do you obtain the necessary compliments to RUP?
10. When performing activities prescribed by RUP, do you perform them as prescribed?
    a. Why do you do them as prescribed?
    b. How do you select what to do and what to skip?
    c. In what ways do you change the activities?
    d. What has the greatest effect on the adaptation of activities?
11. When writing project documentation, do you always use RUP templates?
    a. Which other type of documentation do you use?
    b. Why is other documentation needed?
12. How do you use RUP templates?
    a. Why?

13. Can you use RUP templates exactly as they are?
    a. Do you fill in all information prescribed?
    b. In what ways do you complement RUP templates?
    c. Do you add more information than prescribed?
    d. How do you select which information to skip?
    e. Do you modify the structure of the document?
    f. What has the greatest effect on how you adapt the templates?

More general questions

14. How is RUP related to other methods in use at Volvo IT?
15. Is it suitable to mix RUP with other methods?
16. How do you use RUP?
17. How do you adapt RUP?
18. What has the greatest effect on how RUP is adapted in your organisation?

Final questions:

19. Is there any more information you would like to add which you consider relevant for my investigation?
20. Is it alright if I contact you, e.g. by email, if I need some further clarification of our discussions later?

I would like to thank you for your time and patience answering all my questions.

Have a really nice day ☺

Appendix B – Interview questions

The interview

First I will ask some opening questions to obtain background information. Then I will ask more specific questions, which refer to projects that you have participated in and which have been conducted according to the Rational Unified Process. These questions will be focused separately on the use and adaptation of roles, activities, and templates. For each I will ask following questions such as how and why. After this I will ask more general questions.

Opening questions:

1. How long have you been working at Volvo IT?
2. What have been your major tasks?
3. Which other development methods than RUP have you worked with?

Specific questions:

4. Can you shortly describe the projects using RUP you have participated in?
    a. How many projects using RUP have you participated in?
    b. How many developers participated?
    c. How long were the projects?
    d. Which roles have you had?
    e. What have been your major tasks?
5. In what ways do you adapt roles in RUP?
    a. Why?
6. For the roles you have been assigned, do you perform all activities prescribed by RUP for that role?
    a. How do you select which activities to perform?
    b. What has the greatest effect on how and why these activities are selected?
7. Do the prescriptions in RUP provide sufficient information about how to perform the necessary activities assigned to a role?
    a. Do you add activities not prescribed by RUP?
    b. Where from do you obtain the necessary compliments to RUP?
8. When performing activities prescribed by RUP, do you perform them as prescribed?
    a. Why do you do them as prescribed?
    b. How do you select what to do and what to skip?
    c. In what ways do you change the activities?
    d. What has the greatest effect on how and why activities are adapted?

9. How do you use RUP templates?
    a. Do you fill in all information prescribed?
    b. In what ways do you complement RUP templates?
    c. Do you add more information than prescribed?
    d. How do you select which information to skip?
    e. Do you modify the structure of the document?
    f. What has the greatest effect on how you adapt the templates?
    g. Why?
10. Which other type of documentation do you use other than RUP templates?
    a. Why is other documentation needed?

More general questions

11. How is RUP related to other methods in use at Volvo IT?
12. Is it suitable to mix RUP with other methods?
13. How do you as a developer use and adapt RUP?
14. What has the greatest effect on how and why RUP is adapted in your organisation?

Final questions:

15. Is there any more information you would like to add which you consider relevant for my investigation?
16. Is it alright if I contact you, e.g. by email, if I need some further clarification of our discussions later?

I would like to thank you for your time and patience answering all my questions.

Have a really nice day :-)

Appendix C – Volvo IT mapping of RUP roles

This appendix lists the roles used by Volvo IT. The name of each role is given, which roles from RUP it corresponds, short description, and which RUP artifacts the role is primarily responsible for.

Analyst (System Analyst, Use-Case Specifier & Tester)

The system analyst leads and co-ordinates requirements elicitation and use-case modeling by outlining the system's functionality and delimiting the system.

The use-case specifier details the specification of part of the system's functionality by describing the requirements aspect of one or several use cases. The use-case specifier can also be responsible for a use-case package. The use-case specifier responsible for a use-case package is also responsible for its contained use cases and actors.

The system tester is responsible for executing the system.

Artifacts:

Glossary

Requirements Management Plan

Software Requirements Specification in Combination with Use

Software Requirements Specification

Stakeholder Request

Supplementary Specifications

Use Case Specification

Vision

Architect

The architect leads and coordinates technical activities and artifacts throughout the project. He or she establishes the overall structure for each architectural view: the decomposition of the view, the grouping of elements, and the interfaces between these major groupings.

Artifacts: Software Architect Document

Configuration Manager (Change Control Manager, Configuration Manager & System Integrator)

The configuration manager is responsible for setting up the product structure in the CM system; for defining and allocating workspaces for developers; and for integration. The configuration manager also extracts the appropriate status and metrics reports for the project manager.

Artifacts: Configuration Management Plan

Developer (User-Interface Designer, Capsule Designer, Designer)

The user-interface designer leads and coordinates the prototyping and design of the user interface by capturing requirements on the user interface, including usability requirements; building user-interface prototypes; involving other stakeholders of te user interface, such as end users, in usability reviews and use testing sessions; and reviewing and providing the appropriate feedback on the final implementation of the user interface.

The designer defines the responsibilities, operations, attributes, and relationships of one or several classes and determines how they should be adjusted to the implementation environment. In addition, the designer may have responsibility for one or more design packages or design subsystems, including any classes owned by the packages or subsystems.

Artifacts: Use Case Realization Specification

Implementer (Implementer & Tester)

An implementer is responsible for developing and testing components in accordance with the project's adopted standards so that they can be integrated into larger subsystems. When test components, such as drivers or stubs, must be created to support testing, the implementer is also responsible for developing and testing the test components and corresponding subsystems.

Artifacts: Implementation Model

Process Engineer

The process engineer is responsible for the software development itself. This includes configuring the process before project start-up and continuously improving the process during the development effort.

Artifacts: Development Case

Project Manager (Project Manager & Deployment Manager)

The project manager allocates resources, shapes priorities, coordinates interactions with the customers and users, and generally tries to keep the project team focused on the right goal. The project manager establishes a set of practices that ensure the integrity and quality of project artifacts. The project manager is also responsible for ensuring that there is an effective product change review process.

The deployment manager is responsible for plans to transition the product to the user community. These tasks are documented in deployment plans.

Artifacts:

Iteration Assessment

Iteration Plan

Measurement Plan

Problem Resolution Plan

Product Acceptance Plan

Quality Assurance Plan

Risk List

Risk Management Plan

Software Development Plan

Status Assessment

Bill of Materials

Deployment Plan

Release Notes

Test Designer

The test designer is responsible for the planning, design, implementation, and evaluation of testing, including generation of the test plan and test model, implementation of the test procedures, and evaluation of test coverage, test results, and effectiveness.

Artifacts: Test Plan

Toolsmith

The toolsmith develops tools to support special needs, to provide additional automation of tedious or error prone tasks, and to provide better integration between tools.