BaSyRE: A Lightweight Combination of Proven RE Techniques

Uolevi Nikula
Department of Information Technology
Lappeenranta University of Technology
P.O. Box 20
53851 Lappeenranta, Finland
+358 5 621 2839
Uolevi.Nikula@lut.fi

Jorma Sajaniemi
Department of Computer Science
University of Joensuu
P.O. Box 111
80101 Joensuu, Finland
+358 13 251 7933
Jorma.Sajaniemi@cs.joensuu.fi

Abstract

Many small companies do requirements engineering (RE) superficially or neglect it totally. However, according to software project risk studies a cursorily done RE is one of the biggest risks for software projects. So far RE research has focused on large and complex projects that cannot be managed without proper RE practices. At the same time smaller projects have suffered from basic RE problems. To alleviate this situation we have developed a basic systematic RE method – BaSyRE – that is targeted for small projects developing administrative and business information systems. The key design rationale for BaSyRE are (1) a systematic approach to RE, (2) a ready-to-hand solution, (3) simplicity, and (4) domain specificity. The method itself consists of a requirements document template with techniques and processes for both requirements development and management. All these elements have been adapted to the specific application domain to form a simple but systematic method to be taken in use even without adaptation.

Keywords: requirements engineering, method engineering, lightweight methods, software process improvement

1. Introduction

Superficially done requirements engineering (RE) and especially omission of RE present major risks to any software development project. For example the software project risk identification framework by Keil et al. [1] includes eleven universal risk factors and six of them are related to requirements. Further evidence is provided by the CHAOS survey [2] where the top fourteen causes challenging and impairing projects include six requirements related risk factors only one of which is not mentioned in the Keil et al. study. The combined set of RE related risk factors contains seven issues:

1. Misunderstanding the requirements.
2. Lack of adequate user involvement.
3. Failure to manage end user expectations.
4. Changing scope/objections.
5. Lack of frozen requirements.
6. Conflict between user departments.
7. Incomplete requirements and specifications.

Small companies seem to neglect the advice to follow the good RE practices (see, e.g. [3, 4]) even though there are numerous attempts to ease their adoption. For example method engineering seeks ways to configure methods to the project at hand to better address project specific needs [5], there are books that suggest and explain good RE practices [6, 7], and other books describe comprehensive RE approaches (e.g. [8-11]). However, the success of all these efforts seems to be marginal since, for example, the panels in RE conferences keep repeating the topic, e.g. “Extreme RE: what if there is no time for RE?” [12], “Why is it so difficult to introduce RE research results into mainstream RE practice?” [13], “How can RE research become RE practice?” [14], etc. Considering the risks and the current RE practices, the success in introducing RE into practice seems surprisingly poor. Our interpretation of this situation is that the available methods and especially their adoption require too large investments for small companies.

Small companies often focus on small projects [3] so designing a special method for such projects could provide a solution to this problem. Namely, the complexity of projects tends to increase exponentially with increasing number of components and consequently small projects possess drastically less complexity and interdependencies than large projects. Thus it seems reasonable to expect that lightweight methods that would be easy to adopt and to apply, would prove cost effective for small projects. We believe that companies are ready to adopt a RE method that is ready-to-hand, simple to use, and does not require extensive up front investments. This position is supported by two facts – first in our state-of-

the-practice survey eleven of twelve companies indicated a desire to get help from academia for RE improvement efforts [3]. Second we have started discussions about testing our method in practice and eight of ten companies we contacted indicated interest in testing our method.

Even if we find the simplicity of a method important we also find it necessary to address all the key RE areas simultaneously. A common approach is to divide RE into five key areas – requirements elicitation, analysis, specification, verification, and management [7, 15]. However, we prefer a division into three topics for simplicity: (1) requirements document template, (2) requirements development, and (3) requirements management. In addition to these three topics we find training and tool support for requirements management central for a comprehensive RE method. The need to introduce all these topics simultaneously follows from the notion that all these issues are interrelated and absence of any single element can vitiate the benefits of the other elements. This wide scope of the method leads also to a need to simplify it in some other respects in order to keep it from growing so much that it deters potential users only by its sheer size [16].

Davis [17] has observed that different techniques have been developed or are specifically suitable for particular application domains. Accordingly we limit the variety of techniques in the method by focusing it to a single domain. The application domain of interest to us is small projects developing administrative and business applications. As a limit for small project we consider six to twelve months duration, less than half a dozen workers, and less than a 100 kEUR budget. Another way to characterize the project size is that one requirements engineer suffices for the project. The key properties for administrative and business applications include user interfaces, user data, workflow, and/or other system interfaces. Thus techniques to specify these properties are central to the method.

In this paper we describe a **ba**sic **sy**stematic **RE** method – BaSyRE – that is developed to provide in a ready-to-hand form a lightweight but systematic way to do RE. The method is based on a limited number of existing and proven techniques considering the aforementioned needs. Thus it provides an option for the adopter to simply start using a readymade RE method and modify it later based on own experiences. Since requirements document template development alone could take several months of effort [6] BaSyRE provides a head start in RE process improvement.

The idea of focusing to the key elements of a discipline is not new. Discount usability engineering [18] is an early effort in this direction but it represents rather an idea and its practical implementation than a method. A more recent effort is the extreme programming (XP) which is a clearly a discipline for software engineering [19]. Both these approaches are based on studying the elements of the base discipline in the context of some specific domain and focusing only to the central and proven elements. We have approached RE in the same vein and developed BaSyRE as a lightweight method for small administrative and business information system projects.

This paper is structured as follows. Section 2 reviews the related research in this area. Section 3 explains the design rationale for the method and Section 4 describes the method itself. Section 5 includes the initial evaluation of the method and Section 6 concludes the paper.

2. Related research

Systems development methods have attracted a lot of interest among researchers. For example it has been estimated that there are over thousand brand name methodologies for systems development [20] and a specialized research area called method engineering (ME) has emerged. ME can be divided into four key research areas: meta-modeling techniques, tool interoperability, situational methods, and comparative review of methods and tools [5]. Since we have selected a domain specific approach to RE and further limit the applicability of the method by project characteristics, the situational methods match our approach well. However, the general idea for situational methods is to construct an optimal method for each project separately from method fragments [5] which contradicts our goal of developing a ready-to-hand and simple to use method.

Table 1 presents a survey of 14 RE books and one more general book [21] discussing the Unified Software Development Process and providing a reference point for the other books. We also considered including the Software Engineering Institute's Capability Maturity Model in the review but since it covers RE as mere two RM practices [7] we did not see any benefit in doing it.

The first column in Table 1 contains a reference to the reviewed book and the second one indicates whether the book includes a requirements document (RD) template that can be used as basis for an own RD template. The following four columns focus on requirements development (RDev) and management with techniques and processes for them. For the requirements development techniques we expected the books to present elicitation, analysis, specification, and verification techniques while RM techniques should have addressed change control, version control both for individual requirements and RD, requirements tracing, and requirements statuses [7]. The development process was expected to describe how to use the proposed techniques in a systematic way to produce requirements while RM process should explain how to manage requirements changes. For all these five topics we were interested whether the presentation was accurate enough that a reader is able to try them in practice. The last column indicates whether the book contains a method including a

RD template and covering both development and management aspects of RE. A full compliance with column expectations is marked with a '●' character while partial compliance is marked with a '○' character and a lack of compliance is indicated with a dash.

Table 1. Books reviewed for the method development. A '●' character indicates a full compliance with column expectations, a '○' character partial compliance, and a dash lack of compliance.

Reference	RD Templ	RDev Techn	RDev Process	RM Techn	RM Process	Method
Davis 1993 [17]	●	●	-	-	-	-
Graham 1998 [22]	-	-	●	-	-	-
Hooks & Farry 2000 [23]	-	-	●	○	●	-
Jacobson & al. 1998 [21]	-	-	●	-	-	-
Kotonya & Sommerville 1997 [9]	●	●	●	●	●	○
Kovitz 1998 [24]	●	-	-	-	-	-
Kulak & Guiney 2000 [25]	●	●	●	-	-	-
Lauesen 2002 [26]	●	●	-	-	●	-
Leffingwell & Widrig 1999 [10]	●	●	●	●	●	●
Maciaszek 2001 [27]	●	●	-	○	○	-
McGraw & Harbison 1997 [28]	-	●	●	-	-	-
Robertson & Robertson 1999 [11]	●	●	●	○	-	-
Sommerville & Sawyer 1997 [6]	-	●	●	○	-	-
Wiegers 1999 [7]	●	●	●	●	●	-
Young 2001 [29]	-	-	●	-	-	-

As can be seen in Table 1, few ready-to-hand methods can be found in the literature. The closest match is Leffingwell and Widrig [10] which includes as the last chapter a recipe to get started with RE. The chapter is provided as an answer to a request the authors had often heard in their classrooms: "Just give us a single generic process what we can start with. We know it's not that simple, but we'll be happy to modify it as necessary to *our* project, but we need a more prescriptive starting point, a step-by-step process so that we can better apply what we have learned. *Just tell me how to get started!*" (emphasis original). Consequently Leffingwell and Widrig close their book by suggesting an eight step recipe as an initial process. The prescription for the initial process has a number of assumptions limiting its applicability, e.g. the project size is estimated to be 10-30 team members. Another method close to a complete method is the VORD method described by Kotonya and Sommerville [9]. However, even if the book introduces all the key areas in RE very well, the method itself is limited to requirements development – also suggested by its name VORD (Viewpoint Oriented Requirements Definition).

In summary the reviewed books provide insightful handling of their own areas. However, literature on ready-to-hand methods covering both requirements development and management areas is very limited.

3. Design rationale

The business goal for BaSyRE is to support software development and maintenance with requirements engineering. To achieve this goal we think it is essential to provide a method covering all the key areas in RE in a systematic way in a ready to use form. This approach makes it possible for adopters to first test the method in practice and only after that decide about possible future use of it.

There are two general requirements for the BaSyRE techniques. First the techniques must be proven in practice and second they must be acceptable by the users. The first requirement is addressed in a number of books proposing good RE practices (e.g. [6, 7]). The second one is also addressed in literature – according to [30] the most desirable properties for RE techniques are ease of use, facilitation of good communication, facilitation of capture of complete requirements set, and allowing for or incorporating traceability. BaSyRE has been developed these goals and requirements in mind. The following subsections discuss in detail the specific goals of systematic approach, ready-to-handness, simplicity, and domain specificity.

3.1. A systematic method

A systematic method should cover all the most

important tasks in RE to provide an organized way to perform them. To be able to do this we approach RE from the following four viewpoints:

1. Template for requirements document.
2. Key processes and techniques.
3. Tool support for RM.
4. Training for general RE issues and the method.

These areas should be addressed simultaneously due to their interconnections. For example, adopting only a document template leaves people wonder how to elicit and manage the requirements – issues that are solved by processes and techniques (cf. [6, 7]). Adopting techniques without any training causes confusion and problems; especially knowing the strengths, weaknesses, and limitations of techniques is important before applying them [31]. And finally the use of efficient requirements elicitation techniques and active collaboration with users is likely to lead to so many new requirements and change requests that managing them without tool support becomes unrealistic (cf. [6, 7, 10, 21]).

3.2. A ready-to-hand method

Introna and Whitley [31] analyzed the limits of a method and claim that only ready-to-hand methods are used in practice: "tools are used only if available (ready-to-hand) in the world of doing. If a methodology is not ready-to-hand it will break down and be ignored in the pragmatics of getting the job done". Thus we wanted to develop a method that does not require adoption before usage but provides a quick and simple start on doing RE in a systematic way. In practice this means that

- All required templates are provided in a ready to use form with suggested contents, structure, and layout.
- Key processes and techniques are suggested and explained.
- A tool is suggested to support RM.
- A concise self-study package is available covering both basic RE knowledge and use of the method in practice.

The use of a standard document template is the dominant approach to RD development today (cf. [6, 7, 9-11, 27]). This approach, however, is not without opponents who claim that a prefabricated structure and contents are harmful to the quality of RD [24]. We decided to follow the main stream approach in this question and provide a RD template since it is closer to our ready-to-hand goal. In addition to the contents and structure we also considered the issue of layout. The layout is not a critical issue but having a well-thought layout appears important in the industry and can also improve readability [6]. Thus we adopted a standard office document layout for the templates [32].

The key techniques for RE are explained in detail in many sources (see, e.g. [6, 7, 9-11, 26]). Thus we have been able to just select the most prominent ones from these ready and proven techniques as part of our method. We have embedded these techniques into processes that we have developed or adapted to meet our specific needs. Thus for the method users both techniques and processes are suggested and provided in a ready-to-hand form.

The need to support RM by a tool is acknowledged in many sources (cf. [6, 7, 9-11, 21]). This need is also supported by our discussions with industry even if the needs of companies vary very much. Solutions for these needs range from the use of MS-Word templates to any tool providing benefits over a text editor and to the use of a commercial RM tool. Since we do not have a RM tool designed with the BaSyRE goals in mind we decided to accept all these choices as possible ways to handle RM.

Successful use of a method requires knowledge about both RE in general [31] and the method itself. This knowledge can be obtained in many ways but since even general knowledge of RE seems quite weak in industry [3] it seems justified to provide both kinds of information with the method documentation. For a maximal benefit of information it is important that it is available at the right time which is not true if we rely on users going to libraries to find the appropriate information. Motivation for training, and even possibilities to participate in it, may be limited before the actual need while training after the adoption may simply come too late. Thus the best way to provide the information is a self-study package that can be supplied with the method documentation.

3.3. A simple method

The key to the method simplicity is the synchronization of the requirements development process and the RD template. Our development process includes nine steps and each step suggests the RD topics to address in it. The steps and RD contents are synchronized so that, e.g., the first step addresses only topics in the introduction chapter of the RD. To make the completion of the RD even easier also the participants, suitable techniques, and tasks are suggested for each step. Thus the method includes guidance through the whole requirements development process even though the user is free to follow the process in his/her own discretion.

The second technique to simplify the method is to present all requirements in tables. Our RD template is structured so that there is an own section and table for each requirements category. This table approach is comparable with requirements lists which are known to have various shortcomings (see, e.g. [25]). We chose to use the tables, however, since they provide a consistent and compact way of presenting requirements using concise language – ideas that are advocated by, e.g. [6, 24]. The tables do not automatically result in these benefits but they do have intuitive appeal for these goals which is further supported by providing concise example

requirements for each topic.

Documenting requirements as simple lists works as a first level documentation but this approach is bound to reveal its shortages in the long run. As a solution to this problem our RD template includes second level documentation templates which can be added as appendices to the RD. Thus a user can, e.g., choose to document just the requirement description in the RD or complete also a number of other requirement attributes developed from [7, 11]. Even if the second level approach reduces the simplicity of the method in general, we find it important to demonstrate that many requirements related problems can be solved.

As a final point in simplifying the method we have chosen to limit the method to requirements development and management. In practice this means that all the activities suggested support directly either requirements development or management and nothing else (e.g. collecting measurements [33] or supporting project management [34]). Achieving this goal results in a method that is transparent to the user since it is used to get the job done [31].

3.4. A domain specific method

A domain specific approach is needed to be able to limit the number of techniques included in the method and the size of the RD template. We focus to small projects developing administrative and business applications since both of these domains are fairly well understood which provides a good basis for experimenting with a new method.

The purpose of administrative and business applications is in general to support or enforce a workflow which requires user interaction on data in a data store together with interfaces to user and to other systems. All these topics are standard RD topics so we have just adopted ways to document them from other RD templates: user interfaces [24], dialog maps [7], user data [7], workflow [35], and other system interfaces [36]. In addition to these key topics the RD contents in general were selected from eight different RD templates with the domain properties in mind [7, 10, 24, 36-40]. The reason for considering so many templates is their different foci – e.g. IEEE standard 830 [36] is directed to large projects, Robertsons' Volere template [39] includes extensive project information, and Wiegers [7] and Leffingwell & Widrig [10] have more business focus with a separate vision document. Different sources also provide different templates, checklists, techniques, and processes that are ready to use or close to it.

The key design decision concerning the project size is the focus on small projects where one person can manage RE. This is not a hard limit and does not exclude co-operation but it makes one person responsible for developing and managing the requirements. With this limitation we eliminate the problems associated with multiple persons working simultaneously on the same document. As a consequence, issues that do not require any special attention include inconsistent requirements, contradicting requirements, requirements omissions, simultaneous working on same requirements, and a need for a complex RM tool. It is clear that these questions can be solved but due to our simple application domain their likelihood should be low and addressing them explicitly is unnecessary.

4. BaSyRE method

The BaSyRE method is build around a RD template and the BaSyRE guide. The basic idea is to take the RD template and start filling it in using the requirements development process described in the BaSyRE guide. The BaSyRE guide includes a detailed process description with supporting technique descriptions and checklists. These elements are shown in Figure 1.

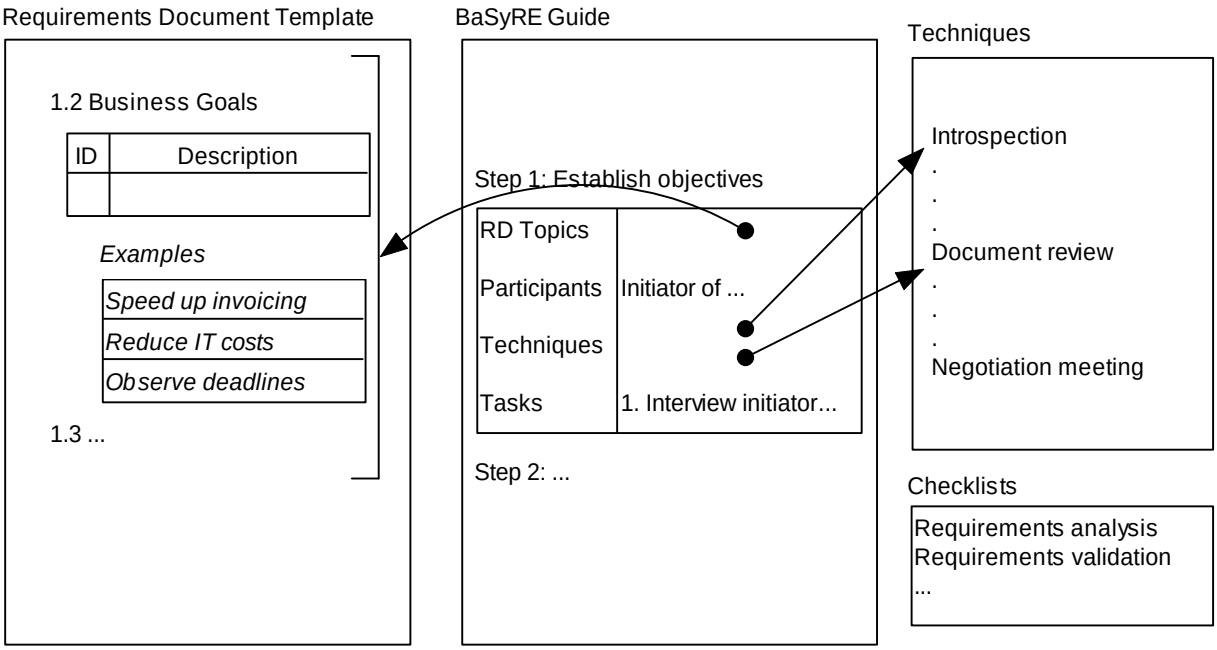

Figure 1. The BaSyRE method.

The RD template covers three main topic areas: introduction, overall description, and specific requirements. The headings for the first and second level topics are shown in Figure 2. The actual RD template includes also third level topics for the cases where more detailed descriptions appear important, e.g. for specific requirements.

1 Introduction
 1.1 Customer Problem
 1.2 Business Goals
 1.3 Stakeholders
 1.4 Product Purpose
 1.5 Product Position Statement
 1.6 Document Overview
 1.7 References
2 Overall Description
 2.1 Product Context
 2.2 Product Perspective
 2.3 Product Functions as Use Cases
 2.4 Users
 2.5 Problem Domain Description
 2.6 System Requirements Allocations
 2.7 Constraints
 2.8 Assumptions
 2.9 Dependencies
3 Specific Requirements
 3.1 Performance Requirements
 3.2 Data Requirements
 3.3 Software System Attributes
 3.4 Other Nonfunctional Requirements
 3.5 Design Constraints
Optional Appendices

Figure 2. The RD contents.

To keep the RD template simple we adopted a two-level approach to it. This means that the term “RD template” refers actually to a set of three templates: RD, interface specification [24], and user manual. The main RD template is also supplemented with five appendices: glossary, typical computer configuration, use case

descriptions, detailed requirements, and rejected requirements. Since such a documentation set may deter inexperienced users, we recommend to use only the main RD template until the role of the other templates becomes obvious.

As Figure 1 shows, each requirement is accompanied by examples to make writing the requirements easier. These examples act as probing hints that help a user to concentrate on important issues and show the appropriate level of accuracy. However, since the need for examples varies a lot they are supplied as hidden text so that the user can choose whether he/she wants to see them or not.

The requirements development process has nine steps that are synchronized with the RD template contents. Figure 3 shows these nine steps and Table 2 describes Step 1 in detail. Presenting the process as consecutive steps gives it a waterfall model like feel even if the process supports evolving requirements in three ways. First the two-level approach suggests the use of RD template for high level requirements (i.e., business goals, context, requirements, and use case lists) while detailed requirements descriptions can be done later as appendixes. Second the process addresses the same topics in different steps, e.g. Step 3 reviews the topics from Steps 1 and 2 to close the business and contextual topics before moving to individual requirements development. The third kind of support follows from the possible use of an iterative software development process. In this case the need to address the high level topics should be considered separately and possibly eliminated from later iterations.

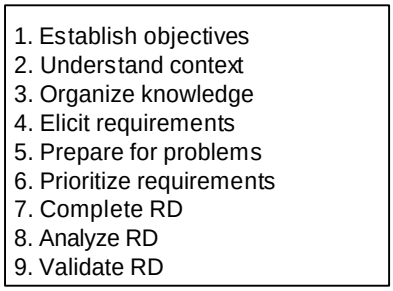

Figure 3. The nine steps in the requirements development process.

Table 2. Requirements development process Step 1: Establish objectives.

RD topics	1.1 Customer problem 1.2 Business goals 1.3 Stakeholders 1.4 Product purpose 1.5 Product position statement
Participants	Initiator of the development effort Stakeholders
Techniques	Introspection Document reviews Electronic requirements Interviews Negotiation meetings
Tasks	1. Interview the initiator to get started with the task and to identify initial stakeholders. 2. Review documents and introspect to get an own idea about the objectives. 3. Interview stakeholders for deeper understanding of the task, use email when appropriate. 4. Do web searches to help develop the product position statement. 5. Arrange meeting(s) with all stakeholders to negotiate about common understanding of the objectives.

As Table 2 shows, the process defines for each step the RD topics to cover and also suggests an order for developing them. Additionally participants, techniques, and tasks are provided to guide an inexperienced user through the development process.

For requirements management BaSyRE provides a set of basic techniques including identification, versioning, baselining, and change management. With these techniques each item can be identified separately and different versions can be differentiated without problems. An evaluation process is suggested to be used before baselining and there is a simple change request process to control the changes. In the case a project has already established change management procedures they should be applied instead of the BaSyRE mechanism. However, basic templates are provided to avoid a need to invent them just for managing requirements changes.

As of now BaSyRE has no special tool support. The need for a simple tool supporting the key design goals of BaSyRE is clear but since different companies have so different needs, we decided to let users select their preferred tool. This includes the use of the BaSyRE document templates using Microsoft Word, continuing

with possible current tool, use of a freeware RM tool (e.g., *sfrm* [41]), and use of some commercial RM tool.

All these topics are described in a self-study package that contains also general RE information. The method specific issues are covered in a BaSyRE guide while general RE information is compiled to a "RE in Short" – document. The latter guide is intended to provide a general introduction to RE and the reasons to do RE properly but it is not intended to replicate existing literature in depth. Thus the guide provides short descriptions of the key topics in simple English and references to more complete descriptions, in most cases to books referred to in Table 1.

5. Evaluation

So far the BaSyRE method has been evaluated based on literature comparisons and industry feedback from our industrial partners. The literature comparisons include three separate evaluations: how would a company applying BaSyRE get rated in REAIMS Top Ten practice evaluation [6], how does BaSyRE address the seven risk factors referred to in the introduction of this paper, and how does BaSyRE compare with the methods described in the related research section.

A standard REAIMS evaluation covers basically all the 66 guidelines proposed in [6] and assesses how they are followed. The options for the assessment are standardized use, normal use, used at the discretion of project manager, and never applied. We find a lighter Top Ten version of this evaluation sufficient in many companies that have not paid special attention to process improvement (cf. [3]). The Top Ten evaluation limits the assessment to the ten practices Sommerville and Sawyer think all organizations should implement [6]. Thus we evaluated BaSyRE by estimating how a company using BaSyRE would score in the REAIMS Top Ten practice evaluation. Calculating the final score accurately is not possible since it requires knowledge of the usage pattern. However, BaSyRE does address all the ten practices by, at least, providing initial versions of the practices. Thus a company adopting BaSyRE as its standard RE method could get even full points in the Top Ten evaluation. However, since BaSyRE is intended to support RE in small projects, some practices do not necessarily ever reach the level required for full points. For example, BaSyRE focuses on providing practical instructions on doing things (i.e., "standards" by [6]) and thus higher-level policies are not directly addressed by it (e.g. guideline "9.2 Define policies for RM"). Similarly guidelines like "6.1 Use language simply, consistently and concisely" depend more on the people writing requirements than any method used to support RE.

We started this paper with seven requirements related project risk factors. These risk factors are listed in Table 3 together with the ways BaSyRE addresses them. In general BaSyRE (1) tries to reduce the likelihood of these risks, and (2) should make the presence of these risks apparent so that appropriate actions can be taken.

Table 3. The requirements related risk factors and the ways BaSyRE addresses them.

Misunderstanding the requirements	Documenting requirements Interacting with the stakeholders (interviews, meetings) Reviewing RD and requirements
Lack of adequate user involvement	Using techniques involving users: interviews, meetings, etc.
Failure to manage end user expectations	Interacting with users (interviews, meetings, etc.) Documenting requirements Prioritizing requirements Validating requirements with prototypes
Changing scope/ objections	Documenting business goals, context, and requirements RM – baselining RD and requirements, change requests
Lack of frozen requirements	See techniques in Changing scope Documenting requirements Use of second level RD for detailed descriptions Change management
Conflict between user departments	Documenting requirements Meetings with all stakeholders Reviewing RD and requirements
Incomplete requirements and specifications	Documenting requirements Use of second level RD for detailed descriptions Requirements analysis techniques Reviewing RD and requirements

Based on our literature survey the methods closest to BaSyRE are the ones proposed by Leffingwell & Widrig [10] and Kotonya & Sommerville [9]. Leffingwell & Widrig address basically all the same aspects of RE as

BaSyRE with few exceptions. The focus of Leffingwell & Widrig is in introducing the whole RE field with techniques and their ready-to-hand method is just a very superficial combination of the proposed elements. For example the RD templates are generic ones proposed for all software projects and include both vision and requirements documents. Thus BaSyRE takes a step further in the ready-to-handness, simplicity, and domain specificity. Especially the requirements development process proposing RD topics to address, participants, techniques, and tasks cannot be found in the Leffingwell & Widrig approach.

Kotonya and Sommerville present the VORD-method in their book, also addressing the whole RE field. The VORD method, however, is focused in requirements development. Thus a key difference is that BaSyRE includes also RM which makes it applicable for the whole RE. In the requirements development phase the differences are quite small. The differentiation factor is actually the level of detail – BaSyRE provides a wide but shallow coverage of different issues while VORD goes much more into the details. For example the viewpoint hierarchies, attributes, and information structures; priority schemes for requirements; event scenarios; conflict identification tables; and even the requirements document template provide much more details than BaSyRE. From our point of view this amount of detailed requirements information seems overwhelming for small projects in administrative and business applications and we suspect how ready projects are to do such accurate specification. Another important difference is the requirements development process which is much more detailed in BaSyRE. However, e.g. conflict analysis is described in much more detail in VORD. As a summary we find BaSyRE approach to requirements development much simpler.

We are currently negotiating with five companies about using the BaSyRE method in practical case studies. In these negotiations we have also asked two questions concerning expected BaSyRE usage: (1) how directly do the companies expect to use BaSyRE and (2) how do they expect to solve the tool issue. Based on the responses two of the companies will start using BaSyRE as such, one will modify it slightly before usage, and two companies will only use suitable parts of the method at first. As tools four companies expect to use of a freeware tool while one company prefers the use of a commercial tool and expects to select one before BaSyRE case studies start. In general the feedback from the method has been positive and the companies have indicated a clear need for such a systematic and ready-to-hand RE method.

6. Conclusion

The BaSyRE method is closing the end of its construction phase and a validation in real projects is needed next. We are currently negotiating about running a number of projects with BaSyRE in five companies. All these companies have indicated interest in testing the method in practice and expect to have projects where BaSyRE can be applied during the fall 2002.

Acknowledgements

The authors would like to thank Heikki Kälviäinen for his comments to an earlier version of this paper.

References

[1] M. Keil, P. E. Cule, K. Lyytinen, and R. C. Schmidt, "A Framework for Identifying Software Project Risks," Communications of the ACM, vol. 41, pp. 76-83, 1998.

[2] The Standish Group, "The CHAOS Report," 1995. Available at www.standishgroup.com. Accessed January 31, 2002.

[3] U. Nikula, J. Sajaniemi, and H. Kälviäinen, "Management View on Current Requirements Engineering Practices in Small and Medium Enterprises," presented at The Fifth Australian Workshop on Requirements Engineering, Queensland University of Technology, Brisbane, Australia, 2000, pp. 81-89.

[4] E. Kamsties, K. Hörmann, and M. Schlich, "Requirements Engineering in Small and Medium Enterprises," Requirements Engineering, vol. 3, pp. 84-90, 1998.

[5] S. Brinkkemper, "Method Engineering: Engineering of Information Systems Development Methods and Tools," Information and Software Technology, vol. 38, pp. 275-280, 1996.

[6] I. Sommerville and P. Sawyer, Requirements Engineering: A Good Practice Guide. Chichester, England: John Wiley & Sons, 1997.

[7] K. E. Wiegers, Software Requirements. Redmond, Washington: Microsoft Press, 1999.

[8] P. Checkland, Systems Thinking, Systems Practice. Chichester: John Wiley & Sons, 1981.

[9] G. Kotonya and I. Sommerville, Requirements Engineering: Processes and Techniques. Chichester: John Wiley & Sons, 1997.

[10] D. Leffingwell and D. Widrig, Managing Software Requirements: A Unified Approach. Reading, Massachusetts: Addison-Wesley, 1999.

[11] S. Robertson and J. Robertson, Mastering the Requirements Process. Harlow, England: Addison-Wesley, 1999.

[12] S. J. Greenspan, "Extreme RE: What If There Is No Time for Requirements Engineering," presented at Fifth IEEE International Symposium on Requirements Engineering, Toronto, Ontario, Canada, 2001, pp. 282-284.

[13] H. Kaindl, "Why Is It So Difficult to Introduce Requirements Engineering Research Results into Mainstream Requirements Engineering Practice," presented at Fourth International Conference on Requirements Engineering, Schaumburg, Illinois, 2000, pp. 67-68.

[14] S. Miller, "How Can Requirements Engineering Research Become Requirements Engineering Practice," presented at Third IEEE International Symposium on Requirements Engineering, Annapolis, Maryland, USA, 1997, pp. 260.

[15] R. H. Thayer and M. Dorfman (eds), Software Requirements Engineering, 2 ed, IEEE Computer Society Press, Los Alamitos, California, 1997.

[16] C. J. Hardy, J. B. Thompson, and H. M. Edwards, "The Use, Limitations and Customization of Structured Systems Development Methods In the United Kingdom," Information and Software Technology, vol. 37, pp. 467-477, 1995.

[17] A. M. Davis, Software Requirements: Objects, States, and Functions: Prentice Hall, 1993.

[18] J. Nielsen, "Applying Discount Usability Engineering," IEEE Software, vol. 12, pp. 98-100, 1995.

[19] K. Beck, Extreme Programming Explained: Embrace Change. Boston: Addison-Wesley, 1999.

[20] B. Fitzgerald, "An Empricial Investigation Into the Adoption Of Systems Development Methodologies," Information and Management, vol. 34, pp. 317-328, 1998.

[21] I. Jacobson, G. Booch, and J. Rumbaugh, The Unified Software Development Process. Reading, Massachusetts: Addison-Wesley, 1998.

[22] I. Graham, Requirements Engineering and Rapid Development: An Object-Oriented Approach. Harlow, England: Addison-Wesley, 1998.

[23] I. F. Hooks and K. A. Farry, Customer-Centered Products: Creating Successful Products Through Smart Requirements Management. New York: AMACOM, 2000.

[24] B. M. Kovitz, Practical Software Requirements: A Manual of Content and Style. Greenwich, England: Manning Publishing Company, 1998.

[25] D. Kulak and E. Guiney, Use Cases: Requirements In Context. Boston: Addison-Wesley, 2000.

[26] S. Lauesen, Software Requirements - Styles and Techniques. Harlow: Addison-Wesley, 2002.

[27] L. A. Maciaszek, Requirements Analysis and System Design: Developing Information Systems with UML. Harlow, England: Addison-Wesley, 2001.

[28] K. McGraw and K. Harbison, User-Centered Requirements: The Scenario-Based Engineering Process. Mahwah, New Jersey: Lawrence Erlbaum Associates, 1997.

[29] R. R. Young, Effective Requirements Practices. Boston: Addison-Wesley, 2001.

[30] C. McPhee and A. Eberlein, "Requirements Engineering for Time-to-Market Projects," presented at Ninth IEEE International Conference and Workshop on the Engineering of Computer-Based Systems, Lund, Sweden, 2002, pp. 17-24.

[31] L. D. Introna and E. A. Whitley, "Against Method-ism: Exploring the Limits Of Method," Logistics Information Management, vol. 10, pp. 235-245, 1997.

[32] Finnish Standards Association SFS, "Layout Of the Document Text Area," in Office Documents. Standards, Finnish Standards Association SFS, Ed., Helsinki: Finnish Standards Association, SFS, 2000, pp. 20.

[33] W. S. Humphrey, A Discipline for Software Engineering. Reading, Massachusetts: Addison-Wesley, 1995.

[34] S. McConnell, Software Project Survival Guide. Redmond, Washington: Microsoft Press, 1998.

[35] J. Cheesman and J. Daniels, UML Components: A Simple Process for Specifying Component-Based Software. Boston: Addison-Wesley, 2000.

[36] ANSI/IEEE Standard 830-1998, "ANSI/IEEE Standard 830-1998," in IEEE Recommended Practice for Software Requirements Specifications, The Institute of Electrical and Electronics Engineers, Ed., New York, NY: IEEE Computer Society Press, 1998.

[37] IEEE/EIA 12207.1-1997, "IEEE/EIA 12207.1, Guide for ISO/IEC 12207, Standard for Information Technology - Software life cycle processes - Life cycle data," in Industry Implementation of International Standard ISO/IEC 12207 : 1995, The Institute of Electrical and Electronics Engineers and Electronic Industries Association Engineering Department, Eds., New York, NY: IEEE Computer Society Press, 1997.

[38] R. S. Pressman, "Adaptable Process Model Document Templates," 2001. Available at http://www.rspa.com/docs/. Accessed March 6, 2002.

[39] S. Robertson and J. Robertson, "Volere Requirements Specification Template, Edition 8," 2001. Available at www.systemsguild.com. Accessed 29 Novemer 2001.

[40] STRÍ TS2, Modelling a Software Quality Handbook, 1 ed. Reykjavik, Iceland: Icelandic Council for Standardization (STRÍ), 1991.

[41] sfrm, "Software for Requirements Management," 2002. Available at http://cs.joensuu.fi/pages/saja/se/sfrm_mainos.html. Accessed June 17, 2002.