JOURNAL OF SOFTWARE MAINTENANCE AND EVOLUTION: RESEARCH AND PRACTICE
J. Softw. Maint. Evol.: Res. Pract. 2010; **22**:53–80
Published online 21 May 2009 in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/spip.420
SOFTWARE PROCESS IMPROVEMENT AND PRACTICE ARTICLE

The impact of agile principles on market-driven software product development

Nina Dzamashvili Fogelström[1,*,†], Tony Gorschek[1]
Mikael Svahnberg[1] and Peo Olsson[2]

[1]*Department of Systems and Software Engineering, Blekinge Institute of Technology, PO Box 520, S-372 25 Ronneby, Sweden*
[2]*Ericsson AB, Business Unit Multimedia Ölandsgatan 1, Box 518 S-371 23 Karlskrona, Sweden*

SUMMARY

Agile development methods such as extreme programming (XP), SCRUM, Lean Software Development (Lean SD) and others have gained much popularity during the last years. Agile methodologies promise faster time-to-market, satisfied customers and high quality software. While these prospects are appealing, the suitability of agile practices to different domains and business contexts still remains unclear. In this article we investigate the applicability of agile principles in the context of market-driven software product development (MDPD), focusing on pre-project activities. This article presents results of a comparison between typical properties of agile methods to the needs of MDPD, as well as findings of a case study conducted at Ericsson, an early adopter of agile product development. The results show misalignment between the agile principles and needs of pre-project activities in market-driven development. This misalignment threatens to subtract from the positive aspects of agile development, but maybe more importantly, threaten the overall product development by disabling effective product management. Copyright © 2009 John Wiley & Sons, Ltd.

KEY WORDS: market-driven software development; software product management; agile methods; case study

1. INTRODUCTION

Agile methods are attractive to software companies since they promise shorter time-to-market, as well as higher flexibility to accommodate changes in the requirements and thereby increased ability to react to changing customer (market) needs (Williams and Cockburn 2003, 2005).

*Correspondence to: Nina Dzamashvili Fogelström, Department of Systems and Software Engineering, Blekinge Institute of Technology, PO Box 520, S-372 25 Ronneby, Sweden.
†E-mail: nino.dzamashvili.fogelstrom@bth.se

The major feature of agile methods is to use iterations of small development projects developing limited sets of the most important functionality at a given point in time. The common property of agile methods is reactive rather than proactive tactics, encouraging quick adaptation to change rather than planning for it. The goal of agile methods is to eliminate long pre-study and analysis of requirements and start feature production as soon as possible. The positive aspect of this approach is that it speeds up the production of functionality that is considered important at present; the downside is that it limits the possibility to control and plan the content of a release (Boehm 2002, Williams and Cockburn 2003).

The flexibility and ability to quickly respond to market fluctuations makes agile/lean development methods attractive for companies operating in a market-driven context, despite the fact that the long-term impact of adopting these principles and their applicability in the market-driven context are to a large extent unknown. Existing studies and experience reports from application of agile methods are mostly isolated to evaluating the performance of these methods on software development activities, such as increasing ädevelopers effiicency, producing better quality code and so on (Dyba and Dingsøyr 2008). Isolated experience articles on increased customer satisfaction exist, but then the studied project was developed according to bespoke practices, i.e. in a customer-developer relationship with a contractually bound effort and not in a market-driven context (Mannaro *et al.*, 2004, Mann and Maurer 2005).

To the best of our knowledge, currently there are no studies analyzing the impacts of agile/lean development principles on either pre-project activities, such as product planning/management in MDPD, or the long-term impact of optimising the project perspective rather than that of products or product portfolios.

The most important aspects of MDPD are requirements triage (initial selection), requirements prioritisation and planning (release planning), i.e. deciding which of the thousands of potential requirements should be implemented, and when (Carlshamre 2002, Karlsson *et al.*, 2003, Regnell *et al.*, 2005). The main task of product management is to weigh all types of criteria ranging from market demands, trends, key-customer demands, to technical and long-term product impact, and arrive at decisions supported by the company and product strategy. Selecting the best sub-set of requirements for implementation has to take both the long-term and short-term perspectives into account as well as market-pull and technology-push.

In light of the effect of pre-project activities on the success of any company practicing market-driven development, it becomes important to know what kind of impact agile principles have on product management and critical pre-project activities such as initial requirements triage, prioritisation, and release planning. This is important in order to determine to what extent agile principles can be adopted in market-driven organisations, and what product management and pre-project practices should be in place to maximise the positive aspects of lean development in projects (Carlshamre 2002, Karlsson *et al.*, 2003, Regnell *et al.*, 2005).

This article investigates the impact of agile principles on pre-project activities in MDPD by studying an industry case. The study is performed in collaboration with Ericsson AB in Sweden, where a large effort to move from 'traditional' pre-study intensive development to lean development (agile) has been ongoing for over 3 years. An early evaluation of the move to lean development and its potential was presented by Tomaszewski *et al.* (2008). The case study in this article presents a postchange evaluation at the same company, but from the perspective of pre-project activities and the potential misalignment between the needs of product management and agile principles. In addition

DOI: 10.1002/spip

to this, the article also qualifies the challenges with future research directions and suggestions as how the misalignment could be handled.

The rest of this article is outlined as follows: Section 2 provides a background to agile software development and market-driven software product development (MDPD) where the common aspects of agile methods are put against the special needs in a market-driven context. Further, a short overview of empirical evidence on agile methods is provided. Section 3 gives details on study design and execution. Case study results, analysis, and discussion can be found in Section 4 and Section 5 respectively. The report is finalised by conclusions and future work in Section 6.

2. BACKGROUND AND RELATED WORK

2.1. Common aspects of agile software development methods

The introduction of extreme programming (XP) in the late 90s is commonly acknowledged as a starting point, coining the term agile in software development. In 2001 other agile initiatives such as Dynamic Systems Development Method (DSDM), Crystal, Feature Driven Development, and SCRUM joined XP and the 'Agile Manifesto' was formed describing the ideas and agile commonalities (Abrahamsson *et al.*, 2003, Cohen *et al.*, 2004).

Nowadays agile methods are quite popular and are seen as an alternative to the traditional plan-driven methods, which thus far have been the basis for accepted practices in software engineering (Abrahamsson *et al.*, 2003, Cohen *et al.*, 2004, Tomaszewski *et al.*, 2008). Plan-driven methods are characterised by heavy upfront planning, focus on documentation, predictability, and repeatable processes (Boehm 2002). Lately these methods have been criticised for their inability to accommodate change and the high costs that are associated with updating documentation and plans (High-smith and Cockburn 2001, Poppendieck and Poppendieck 2003, Leffingwell 2007). On the other hand, agile methods are designed to accommodate rapid change (Beck 2000, Williams and Cockburn 2003). In order to be flexible, agile methods rely on individuals and their knowledge rather than processes, value working piece of software over complete requirements specifications and design plans; and encourage responding to a change rather than planning for it (Manifesto for Agile Software Development 2008).

The latest contribution to the agile community, Lean Software Development (Lean SD) (Poppendieck and Poppendieck 2003), is based on a collection of attitudes and principles originating from the successful concepts of lean manufacturing developed by Toyota known as the Toyota production system (Womack *et al.*, 1991). The original lean manufacturing principles at Toyota (Eliminate waste, Amplify learning, Decide as late as possible, Deliver as fast as possible, Empower the team, Build integrity, etc.) are based on continuous improvement and elimination of waste. Since these principles are rather general they can be interpreted and adapted to suit different applications and business domains, for example software development. This can be seen as an advantage as the applicability of agile methods are generally considered to be limited to small- to medium-sized enterprise developing medium-sized non-critical applications (Boehm 2002, Williams and Cockburn 2003, Sillitti *et al.*, 2005). However, the current translation of lean principles to software development (Poppendieck and Poppendieck

DOI: 10.1002/spip

2003) is fully compliant with, or even designed after, existing agile principles. This might imply a limitation of this potential, limiting the flexibility of the origin ideas behind the lean development.

In general, agile methods may vary when it comes to the specific application of certain practices. However, they all follow the main principle of producing working software in small iterations, and utilising small teams of dedicated individuals that focus on software development. The common aspects of agile methods can be summarised by the following principles [hereafter called Agile Property (AP)] (Highsmith and Cockburn 2001, Abrahamsson *et al.*, 2003, Cohen *et al.*, 2004):

- **AP1. Feature orientation**: The main focus is on the production of features as soon as possible. The goal is to deliver working functionality that is perceived to have the most value for the customer, and this is done in small and frequent iterations.
- **AP2. Reactive development**: Reactive development is about responding to a change instead of planning ahead, and delaying decisions as long as possible. Agile methods promise flexibility and follow the ideology that one can not control the world, thus the strategy is to respond to a change rather than plan for it. Examples of reactive practices are refactoring, adjusting requirements priorities, and release scope after each iteration and delaying decisions for as long as possible.
- **AP3. Evolving project (release) scope**: Perhaps one of the main distinguishing features of agile development is the change from a fix-scope approach to a more open-ended approach. In the traditional fix-scope approach much effort is spent on defining and planning the content of a product release of a project upfront. In agile the release scope is emerging in the process of development rather than planned ahead. A prioritised list of requirements (product backlog in SCRUM and prioritised user stories in XP) serves as an initial input which is a kind of wish-list of a release scope. The release scope is expected to be refined and updated at the end of each iteration in order to accommodate changes and new information that was learned in the latest iteration. This practice is in line with the reactive development property described in AP2.

2.2. Specific needs of MDPD

In MDPD software products are often developed as product families that are offered to a mass market rather than a specific customer. The product development has an iterative character where new versions of the product are delivered to the market via established release cycles (Dahlstedt *et al.*, 2003, Gorschek and Wohlin 2006).

The fact that the product is not ordered and developed to suit the wishes and needs of one specific customer creates special needs distinguishing the characteristics of this type of development from customer-specific (bespoke development) (Potts 1995). In a bespoke situation the development organisation can rely on the customer to decide what functionality should be included in the product and this information can be elicited directly from the customer. However, in market-driven development this is not feasible as there are any number of potential customers. Instead the development organisation itself decides what functionality should be delivered to what market

DOI: 10.1002/spip

segment and when (Potts 1995, Karlsson *et al.*, 2003, Ebert 2005, Regnell *et al.*, 2005, van de Weerd *et al.*, 2006). This creates a special focus on pre-project activities usually associated with product planning and management operations (van de Weerd *et al.*, 2006). The specific needs of MDPD are summarised below:

- **N1. Balancing different requirements types**: Selecting a set of requirements that will be included in future releases of a product is one of the most important strategic decisions in MDPD. This decision is based on comprehensive analysis of the company's business intelligence, product planning and road-mapping, marketing and sales information, and strategic assets such as engineering know-how and investments in product architecture (Greer and Ruhe 2004, Ebert 2005, Regnell *et al.*, 2005, Saliu and Ruhe 2005). The specifics of this task require that the development organisation must find an optimal balance between different requirement types such as commercial requirements, innovations, requirements connected to architecture improvements and also balance quality attributes like maintainability, performance, scalability, and usability across different releases.
- **N2. Trade-off between market-pull and technology-push**: Studies in relation to MDPD have shown that the development organisation not only needs to respond to the current consumer needs but also anticipate future needs as well as influence the needs of the consumers via innovations and technology breakthrough. Finding a balance between market-pull and technology-push is considered one of the challenging aspects for companies operating in this context as it indirectly involves balancing low- and high-risk as well as long- and short-term considerations (Karlsson *et al.*, 2003, Regnell *et al.*, 2005).
- **N3. Release content planning (requirements selection)**: Planning the content of a specific release is recognised as one of the most challenging parts of MDPD (Carlshamre *et al.*, 2001, Carlshamre 2002, Greer and Ruhe 2004). This activity occurs after a large number of initial requirements have been filtered through the steps of requirements triage/screening and prioritisation. The reduced amount of requirements is then considered for inclusion for a specific release, which is the task of release scope planning. The ultimate goal of this activity is to find a selection of requirements that will give the best possible return on investment (ROI) and will be feasible to implement within a given time- and resource-frame. The main characteristics of this task require finding an appropriate balance between relative cost and value of each requirement, technical and business dependencies between requirements, and available budget and resources at hand. The outcome of this activity results in a list of requirements that constitutes the planned scope of a release. Having an established release scope is important from three main aspects. One, it enables decision makers to estimate and plan for the total value of a release, and a comparison to the total value needed for a specific market segment at a certain time. The goals of a release may be combinatory in nature, i.e. several projects in collaboration over time can together offer a specific value for the market. Two, a set release scope provides a baseline that can be used to effectively evaluate emerging changes with respect to how this change affects the total value and cost of a release. This provides a basis for informed decisions and helps managers to not only react but also find the best way to react on a given change. Three, it provides a possibility to plan and implement long-term product development and innovation that are not based on current customer wishes but focus on, for example, system evolution goals that go beyond the project perspective.

DOI: 10.1002/spip

Table I. Agile properties vs. MDPD needs.

Agile property	Affected MDPD need
AP1. Feature orientation	N1. Balancing different requirements types
AP2. Reactive development	N2. Trade-off between market-pull and technology-push
AP3. Evolving release scope	N3. Release content planning

2.3. Applying agile principles in MDPD

In order to understand the possible impact of agile methods on pre-project activities, the properties of agile methods described in Section 2.1 are put against the needs of MDPD described in Section 2.2 The result is presented in Table I.

In Table I, AP1 is put against N1 since feature orientation by definition favors feature-type requirements thereby being the opposite of creating a balance between different requirement types such as feature and non-functional requirements. AP2: Reactive development is associated with N2, i.e. the need of finding trade-off between market-pull and technology-push since there is a risk that the agile view of adopting and reacting to customer needs results in the company becoming too focused on responding to current market needs and thus being unable to push innovative ideas that do not directly originate from the market or key-customers at present. This can have devastating effects on the long-term, invisible to the project perspective which is inherently focused on delivering the requirements relevant from the project perspective.

Using an evolving release scope (AP3) is by definition at odds with planning the release scope (N3) as it limits the decision maker's ability to actively control and influence release content. As presented in Section 2.2 in MDPD there is a need for an established release scope, where the decision on what is included in a release is based on careful analysis of all candidate requirements (Carlshamre 2002, Greer and Ruhe 2004, Saliu and Ruhe 2005, Ngo-The and Ruhe 2008). In contrast, an agile project release scope is loosely defined and is based on rough estimates and the initial feelings of the customer about the requirements. The intention in an agile project is not to have the best selection but to refine the scope during the course of development (Beck and Fowler 2001, Poppendieck and Poppendieck 2003, Williams and Cockburn 2003, Leffingwell 2007).

The lack of alignment between the properties of agile methods and the needs of MDPD can be further explored by studying certain assumptions made in agile development methods and mapping them to the product development practices in market-driven context. These assumptions are listed in Table II. Each is elaborated upon below.

- **Development context**: Agile methods are bespoke in origin (Karlstrom and Runeson 2005) and largely focus on a specific project rather than long-term product development common to MDPD where the focus is on the product or product families offered to any number of potential customers. Success in a bespoke (project) perspective is the successful completion of the project fulfilling the needs of the customer. Success from a market-driven perspective is measured in sales, revenue, market growth, and the ability to create and maintain a flexible product architecture that will support future product releases. Thus the difference between agile and MDPD context is both in the scale and the complexity of the projects. For example creating a maintainable product architecture is not as critical in a small bespoke project compared to

DOI: 10.1002/spip

Table II. Agile assumptions and MDPD reality.

	Agile methods	MDPD
Development context	One customer focus on a specific project. The project is central	Many potential customers focus on a product or product family. Development is focused on the product, and any number of projects can contribute to a release
Business context	Customer owns the product and finances the development effort	Development organisation owns the product and finances the development activities itself
	The release scope can be undefined and negotiated	The release scope needs to be defined and is not easily renegotiable
Understanding of value	Simple	Complex
Assumptions about requirements	Requirements are at large unknown	Requirements are known through market investigations and business intelligence

a project that is part of a large-scale product development effort of a product that will have a long life span.

- **Business context**: Agile methods assume that the customer is responsible for identifying valuable requirements and financing development activities, as well as that the scope of the release or a project is negotiable. In contrast in MDPD the development organisation itself is responsible and has to decide what requirements are valuable for a current release and for finding financial resources to develop the selected set of functionality. The contents of the release can not be negotiable in the same sense as in agile projects as there is no specific customer, rather the decision of what to develop is arrived at after complex deliberation involving product management, marketing, key-customers, sales and upper management, as well as long-term plan influence. In fact, the scope of the release has to be defined since it is quite common that product management need to produce a marketing message on what is to be expected in the next release of the product long before the development activities are finished (or even initiated). Applying the concept of evolving release scope has some economical consequences as well where the developing organisation (which in MDPD stands for the development costs itself) must evaluate how much updating, refactoring and other correcting after-the-fact activities can be afforded by a company that has to deliver a certain value at a defined market window time-point. In MDPD time-to-market is generally seen as being of paramount importance.
- **Understanding of value**: Agile methods focus on creating rapid value for a specific customer. In MDPD the definition of what constitutes value is more complex and the potential value for a specific customer is just one component of a value measure (this relates to N1 and N2 presented in Section 2.2). Also, agile methods assume that the value of rapidly developed functionality can be confirmed by a customer or an on-site customer representative almost immediately. In MDPD the validation of perceived value of developed functionality requires longer lead-times as well as different tools, for example GAP analysis, customer value analysis (CVA) and focus groups (Gorschek 2006). An agile project member cannot go to an on-site customer as this representative does not exist. The concept of value not only transcends any one customer but also the project itself. For example, a project can be a part of a long-term development plan

DOI: 10.1002/spip

to offer infrastructure to subsequent development down the line. From a project perspective most of the project contents might seem as waste, but from a product perspective the view is another (Gorschek and Davis 2008).

- **Assumptions about requirements**: Agile methods assume that at the beginning of the development project requirements are largely unknown. This is not the case for MDPD where large efforts have been devoted to collecting large amounts of potential requirements, which undergo triage, and subsequent refinement and selection (and packaging) prior to the project being created (Potts 1995, Karlsson *et al.*, 2003, Ebert 2005, Reg-nell *et al.*, 2005, van de Weerd *et al.*, 2006). This does not mean that the requirements given to a project are perfect or even complete, but it does mean that there has been a deliberate and critical effort to select and deliver a set of requirements to the project.

As a consequence to these differences in assumptions and properties of agile development methods and MDPD needs, it seems reasonable to assume that applying agile principles in this context will have certain consequences on pre-project activities connected to requirements selection and product planning. This is the focus of this article and the case study presented in Sections 3 and 4.

First a background to agile development is given with focus on reporting experiences from industry usage.

2.4. Empirical studies on agile methods

Agile software development models are relatively new and have not been tested to the same extent as traditional software development models (Abrahamsson *et al.*, 2003). A large part of the existing empirical evidence regarding agile methods is focused on XP. Here research efforts are mainly focused on studying isolated practices, such as pair programming or test-driven development (Erickson *et al.*, 2005). When it comes to the impact of agile methods on software development activities a number of case studies exist. These studies report improvements in the quality of the developed code, increased effectiveness and productivity of software developers, and customer satisfaction (Middleton 2001, Ilieva *et al.*, 2004, Layman *et al.*, 2004, Man-naro *et al.*, 2004, Mann and Maurer 2005). However, a closer examination of these studies shows that most of them focus on identifying the effect of agile principles on in-project activities, and that observed effects are primarily identified in the context of small development teams developing small- to medium-sized applications. These results are in line with the findings of Abrahamsson *et al.* (2003), who in their survey concluded that there is a lack of sound empirical evidence to support the applicability of agile methods in different development contexts and application domains.

Empirical evidence in relation to the application of agile practices in large-scale contexts is quite scarce. In a recently conducted systematic review of empirical studies on agile software development 270 reports were identified as containing empirical results on agile development methods (Dyba and Dingsøyr 2008). After assessment of these according to the principles of what constitutes good practices of empirical research, only 33 of the original 270 were considered as credible empirical reports. Out of these only a small fraction, two reports (Dagnino *et al.*, 2004, Karlstrom and Runeson 2005), report on application of agile principles in large organisations. Both of these studies focus on in-project activities and report positive results of applying agile principles. It is important to notice though that in the works of Karlstrom and Runeson (2005) the application of XP practices was done in the context of small pilot teams. Information about the context of development size

DOI: 10.1002/spip

of developing project and which XP practices had been applied was unfortunately missing. In the works of Dagnino *et al.* (2004) a rather conservative version of agile development methodology was used to produce a prototype for an internal customer.

2.4.1. Empirical studies on the impact of agile methods on MDPD

When reviewing existing knowledge and research on agile methods it is easy to notice that most of the empirical reports focus on analyzing the impact of agile methods on in-project activities such as software design, development and test activities, when studies focusing on pre-project activities and product management in general are quite rare (see Section 2.4).

Recently there has been interest in investigating the application of agile methods for product line engineering (PLE) (Clements and Northrop 2002), which have some similarities with product management activities. At the current stage however the research in this area is quite new, focusing on studying in what way agile principles and PLE activities can be combined (Tian and Cooper 2006, Hanssen and Fægri 2008), or the application of the selected principles of agile methods (such as collaboration and focus on individuals) in some PLE activities (Noor *et al.*, 2008).

When it comes to MDPD, to the best of our knowledge and for the time being, studies focusing on understanding of the impact of agile principles on pre-project, product management activities do not exist. This situation indicates a critical gap in understanding, e.g. the long-term effects of agile methods on product development. This is especially relevant when considering MDPD, where pre-project activities are critical (Regnell *et al.*, 2005).

Studies investigating the impact of agile principles on pre-project activities are needed in order to determine to what extent agile principles can be adopted in the MDPD context. This can involve e.g. how agile models need to be tailored, or what other practices need to be in place in order to take full advantage of positive aspects of agile development without negatively effecting MDPD.

The case study described in this article investigates the characteristics of decision making in Ericsson AB that has worked with adopting agile principles over the last years. The goal of the case study is to study and understand the characteristics of the pre-project decision-making process when applying agile development principles. The identified characteristics and specifics of the pre-project process are analyzed in order to see if the study findings can support the expected impact of agile principles on pre-project activities in MDPD as described in Section 2.3.

3. CASE STUDY SETTING AND DESIGN

The case study was conducted during the fall of 2007 at Ericsson AB in Sweden (called Ericsson hereafter). Ericsson is one of the world's leading companies in telecommunication, providing a wide range of products and solutions. The company operates in a market-driven context where the products are sold as generic solutions offered to an open market, although customised versions of the products are also developed and sold to key-customers.

With the goals of achieving higher process performance and increased flexibility when it comes to accommodating changing requirements Ericsson moved from a traditional development process with relatively extensive development projects which lasted about a year or even more (following the classical waterfall development steps) to lean development. One of the main motivations for moving

DOI: 10.1002/spip

to a new development process was to minimise the number of change requests. Change requests were associated with high development costs since they often resulted in wasting development effort in form of detailed analysis of requirements that were not implemented in the final product or expenses associated with rework.

According to Tomaszewski *et al.* (2008), the traditional waterfall development projects at Ericsson suffered from large number of change requests mainly caused by long development cycles and extensive analysis of the pre-defined release project scope. The new agile development process called Streamline (Tomaszewski *et al.*, 2008) promised to cut down the costs associated with the change requests substantially. A recent evaluation of the application of Streamline at Ericsson reports reduction of requirements volatility in the projects mainly caused by application of small development packages and shortening the lead-time of development projects (Petersen and Wohlin 2008). The Streamline development process (called only Streamline hereafter) is strongly influenced by agile and especially Lean SD, and a summary of it can be found in Section 4.1.

3.1. Research roadmap

The goal of this study was to identify how the introduction of Streamline has affected product management [pre-project requirements engineering (RE)] activities practiced at Ericsson. In order to achieve this goal it was decided to first identify and study the process and the characteristics of pre-project decisions at Ericsson. Collected information was then analyzed using the structure seen in Table I. The intention here was to identify how well the study findings support the anticipated impact of agile principles on MDPD outlined in Section 2.3.

Information about the practiced pre-project activities in Streamline was gathered by means of studying the pre-project RE decision-making process applied in the development of three separate products at Ericsson. Product 1 and product 2 are large mature systems with a large number of releases operating all over the world at many customer sites. Products 1 and 2 are over 9 years old and have undergone seven and four releases respectively. Product 3 is a smaller product with only one previous release since its creation in 2006.

In order to get an overview of the RE decision-making process three main sources were used to achieve a triangulation (Gorschek and Wohlin 2004) of sources as can be seen in Figure 1.

Part A consisted of the study of official process documentation, Part B was interviews with practitioners allowing for a balanced view where the official process was weighed against the one actually practiced, and Part C consisted of observations (e.g. sitting at the meetings). The use and

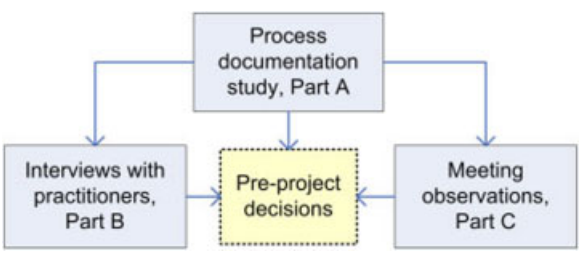

Figure 1. Study overview of mapping pre-project RE decisions.

DOI: 10.1002/spip

comparison between different sources is at the central core of the triangulation (Gorschek and Wohlin 2004, Pettersson *et al.*, 2008). The details of study execution are provided in Section 3.2, and the results obtained are detailed in Section 4.

3.2. Study execution

The initial stage was focused on studying the official process documentation in order to identify pre-project decisions, roles and responsibilities associated with the decisions, and mapping the decision characteristics. In addition, the study of the official process allowed a mapping of meetings and forums for decision-making.

3.2.1. Process documentation study (part A)

The documentation study was conducted by means of reviewing official process descriptions for pre-project activities in Streamline. A senior expert, responsible for process improvement activities was used as a guide to the formal process in order to identify relevant process documents. An important selection criterion for the document study was to consider those process descriptions and workflow guidelines that were common for development activities.

The information provided in the process documentation was analyzed in order to identify the major steps/activities and the decisions points involved in pre-project RE. For each identified decision point the characteristics were mapped. Characteristics in this case are the (i) goal(s) for an activity, (ii) the decision criteria/method used, and (iii) the decision support material used and/or produced. The identified roles and decision forums were later used to plan the interviews and observations.

3.2.2. Interviews with practitioners (Part B)

The intention with the interviews was to elicit information on the pre-project decision-making process not evident from the formal process descriptions in Part A. Semi-structured interviews were used in an informal setting.

At the beginning of each interview the participant was asked to describe the process of working with pre-project requirements. In most cases the interviewees would draw a figure of the overall process and map the major steps/decisions in the process.

Once the major steps and decisions were identified the interview continued detailing them and the participants were asked a set of questions designed to retrieve the characteristics of the decision points. In order to allow a comparison between the formal and practiced process, for each mentioned decision, the interviewees answered questions on goals of a decision, how the decisions were taken, what information was important/necessary in order to take a decision (decision support material used) as outlined in Section 3.2.1. At the end of each interview participants were asked to identify major challenges and improvement suggestions in relation to the process and the characteristics.

The results of the interview were transcribed into session summary sheets (Robson 2002) directly after the interview session. Information placed in these sheets followed the same template, providing data on interviewed person, issues covered, and researcher's own notes regarding the information obtained in the interview.

DOI: 10.1002/spip

Table III. Roles identified in pre-project RE.

Name (number interviewed)	Description	Responsibility
Product Manager (5)	This role has the strategic product responsibility and decides the overall product development direction	Select, prioritise and plan requirements for future releases
System Expert (3)	This role is populated by people with expert knowledge of the systems and their architecture System experts provide decision material for product management since they are responsible for providing analysis of pre-project requirements, in the form of feasibility, impact and technical dependencies	Responsibility of system architecture and system development plan
Line Manager (3)	Line managers have overall responsibility of Development and Testing (R&D) resources. They plan for R&D utilisation in development projects and allocate resources to develop requirements selected by product management	R&D capacity overview, planning of development projects
Project Manager (3)	Responsible for the detailed planning, execution and follow-up of development projects for a specific release	Planning and execution of a specific project

Once the interviews were completed, the information from the individual interview was coded in different data categories related to the practiced pre-project requirements decisions and their characteristics. The utilised data categories covered topics such as practices requirements decision steps, requirement types considered at each decision, practiced approach for taking decision, required decision material, experienced challenges, etc. The coded data from each interview was then placed in an excel sheet where it was analyzed again in order to find relations to the agile properties and MDPD needs presented in Table I.

The selection of the interview participants was based on the process documentation where roles involved in pre-project RE and decision-making were identified.

Further, the case study consisted of the study of three different products at Ericsson to make sure that the process and the characteristics studied were representative for the organisation. Therefore the interviewees were also selected to represent the important roles for each of the products (see Table III).

In total 14 interviews were conducted. Each interview was conducted individually. To avoid validity issues such as the subjects feeling uncomfortable with speaking their mind none of the interviews were recorded. Detailed notes were taken during the interviews, and during the subsequent refinement of these notes supplemental questions were posed to the interview subjects post-interview if needed. The detailed and refined notes were then used to validate understanding and avoid misinterpretations as they were shared with the subjects for validation purposes. The distribution of the

DOI: 10.1002/spip

Table IV. Decision forums.

Name (number visited)	Purpose
Pre-project Requirements Council (2)	This council consists of strategic product managers, technical product managers and system experts. Decision forum is used for decisions on which requirements will be approved for further analysis and how the selected requirements will be allocated.
Prioritisation Workshop	Workshop participants include product managers, system experts and different requirement representatives. Decisions on which requirements will be assigned highest priority for the next release are taken.
Release Scope Refinement Board (1)	The board consists of product manager, line manager and system experts. Here the decisions on approving implementation proposals and associated cost of a requirement are taken.
Release Planning Board (1)	The board consists of product manager, line manager, system expert and project manager. Here the decisions regarding release scope planning and initiating of development projects are taken.

interviews was the following: Product Managers – 5; System Experts – 3; Line Managers – 3; Project Managers – 3.

3.2.3. Observations (part C)

This part of the study was focused on attending forums and meetings where pre-project decisions were discussed. These forums were identified initially in the process study (Part A) and later confirmed by the process expert at the company. Identified decision forums are summarised in Table IV.

In total four meetings were attended; Pre-project Requirements Council (2), Release Scope refinement Board (1) and Release Planning Board (1). In order to minimise any potential effect caused by the presence of a researcher the meetings were not recorded. Instead a researcher observed the character and content of discussions and decision-making process during the meetings and notes were taken.

4. EMPIRICAL RESULTS

4.1. Process documentation study (part A)

Figure 2 presents the major pre-project activities in Streamline as described in the process documentation. The activities are displayed in chronological order (order of execution in the company).

DOI: 10.1002/spip

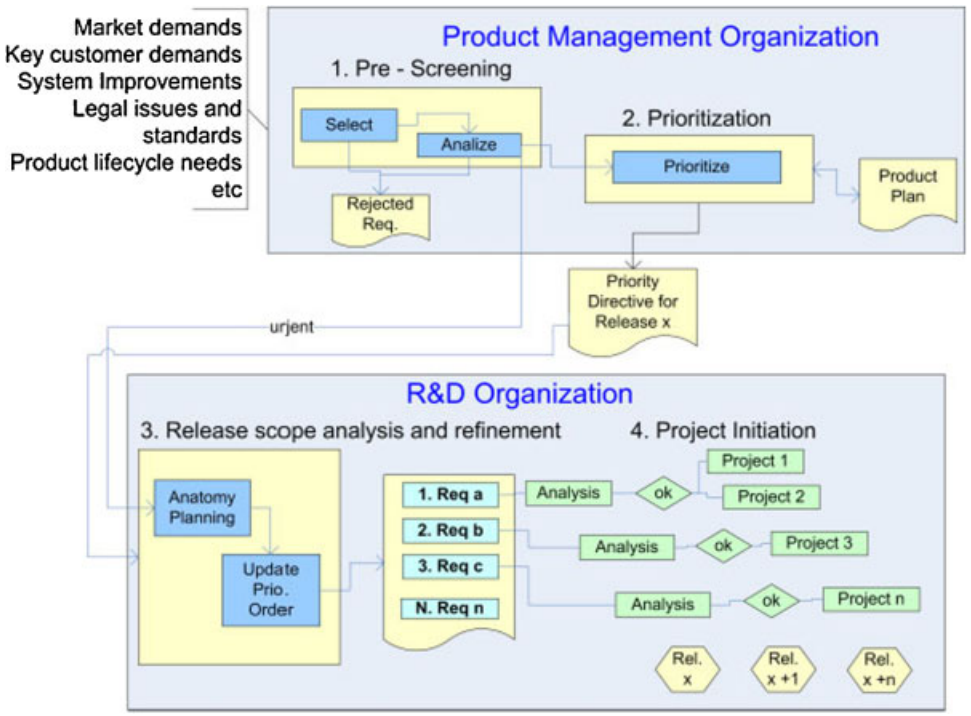

Figure 2. Pre-project decision making process in Streamline.

As shown in the figure, pre-screening of incoming requirements, requirements prioritisation and release scope analysis and refinement are the major activities performed prior to initiating development projects.

4.1.1. Pre-screening

According to Streamline, initial requirements are elicited from different sources in a continuous manner. Incoming requirements are then inserted in the repository after which they pass through the pre-screening step. The goal of this activity is to identify a sub-set of requirements that is interesting to include in a certain product. The process involves two sub-steps: selection and analysis. During selection requirements are analyzed in order to determine suitability of a requirement for a certain product. In this step requirements that are wrongly allocated to a product, duplicate requirements and other requirements that are considered not suitable for a product are filtered away. During analysis requirements are analyzed to determine their business value and technical impact. This information together with directives from product plans and roadmaps are used to allocate requirement to a certain release of a product. At the end of the pre-screening step an initial requirement is either allocated to a future release, rejected, or considered so urgent that is allocated to the release under definition.

DOI: 10.1002/spip

4.1.2. *Prioritisation*

During this step requirements that are allocated to a certain release are prioritised in order to produce an initial release scope. At the end of a prioritisation step only a small sub-set of all requirements that are allocated to a certain release are selected to be included in the list of prioritised requirements for a certain release which in Streamline is called 'Priority directive' of a release. At the end of the prioritisation step requirements included in the 'Priority directive' are handed over to the R&D organisation (development and testing) for analysis and realisation.

4.1.3. *Release scope analysis and refinement*

The goal of this activity is to analyze and package requirements included in the 'Priority directive' into small development projects each in size approximately 3 months. Activities in this step have a continuous and iterative character where requirements are analyzed and assigned to development projects following the order of their priority in the 'Priority directive'.

Once the priority directive is received by R&D the initial step is to create a project anatomy plan. The purpose of the anatomy plan is to identify and describe functional dependencies between the requirements in the priority directive, which are subsequently used to update the priority order taking the dependencies into account, resulting in a function container list. This list provides a base for the planning of development projects as it dictates in which order the requirements will be analyzed and assigned to the development projects.

When the first version of the function container list is available, R&D can start conducting technical analysis of requirements following the priority order dictated by the list. Requirements are analyzed independently and development projects are initiated as soon as the line management in the R&D organisation can find and allocate resources necessary for realisation of the selected solution proposal.

The technical analysis of each requirement should take a maximum of twenty (20) person-hours and produce a report detailing which components of the existing system will be affected by the new functionality, possible implementation solutions, and a cost estimate of each solution. The results of the technical investigation is presented at the Release Planning Board (see Table IV) and the solution that provides the best trade-off between product management priorities, resource capacity, and system architecture impact is selected. If an acceptable trade-off cannot be reached, for example the estimated cost of requirements implementation is higher than expected by product management, or the resource capacity is inadequate the priority order of this requirement must be renegotiated.

4.1.4. *Summary*

Streamline is a process that tries to apply agile principles in an MDPD context. The process has clear characteristics of market-driven development, where requirements are collected from different sources and filtered through the pre-screening and prioritisation steps to produce the 'Priority directive' of a release.

Streamline also has a strong connection to agile and lean development principles, which is evident in the step of release content analysis and refinement. As shown in Section 4.1.3, the scope of

DOI: 10.1002/spip

Table V. Practiced approach for pre-screening.

	Approach/criteria for pre-screening	Reqirement types
Product 1 and Product 2	Practiced approach for pre-screening is focused on identifying: (i) duplicates, (ii) requirements that are wrongly addressed (not applicable for a product), (iii) unreasonable requirements like you should have no downtime	Commercial requirements from influential customers and market units
Product 3	Focus on evaluating impact and cost of a requirement	Commercial requirements from influential customers and market units

a release if not fixed and development activities do not wait until analysis of all requirements allocated to a release are finalised, instead development projects are initiated as soon as analysis of top priority requirement allows for identifying what resources are necessary and these resources can be allocated. This approach is in line with the agile property AP3 - Evolving release scope. In Streamline this property is intended to ensure that the development effort is focused on top priority requirements and to minimise waste connected to change requests mentioned in Section 3.

4.2. Interviews and observations (part B and part C)

In this section we describe study results obtained via interviews with the practitioners and conducted meeting observations. The results are grouped according to major pre-project steps identified in the process documentation study (see Section 4.1).

4.2.1. Results for Pre-screening (Step 1 in Figure 2)

4.2.1.1. Finding 1: Practiced Approach for Pre-screening . Interviews with the practitioners showed that in general practiced activities followed the steps of the documented process. However some differences were identified.

As shown in Table V, for product 1 and product 2 activities in pre-screening mainly focus on defining if a requirement is appropriate for the product, rather than evaluating requirement's business value and technical impact as prescribed by the process documentation. This type of analysis was performed in an informal way and no reports were produced. Product managers for this product explained: '*we receive hundreds of requirements; therefore the first step is to lower the volume by removing duplicates and requirements that are wrongly allocated to our product*'.

In the case of product 3 the pre-project activities considered both technical analysis and value evaluation. However compared to the technical analysis, evaluation of the requirements value received little attention. Moreover, mechanisms used for evaluating and determining value of a requirement were not obvious. Observation of Pre-Project Requirements Council meetings for product 3 confirmed this since most of the meeting time was spent on clarifying technical details around the requirements. Technical evaluation was mostly based on expert opinion. For requirements that were considered complex an official technical report was ordered and produced.

DOI: 10.1002/spip

4.2.1.2. Finding 2: Balance between Different Requirement Types at Pre-screening During the interviews product managers of each product were asked to list the different types of the requirements considered by them during pre-screening. All product managers listed commercial requirements originating mainly from influential customers and marketing units, while other types of requirements like technical and architectural improvements or product life-cycle requirements were not mentioned.

4.2.2. Results for requirements prioritisation (Step 2 in Figure 2)

When it comes to requirements prioritisation, interviews with practitioners provided detailed information on the applied prioritisation technique and evaluation criteria, which was missing in the formal process descriptions. Findings identified through analysis of interview results are presented below:

4.2.2.1. Finding 3: Practiced Approach for Prioritisation. For all the products studied the requirements prioritisation of requirements in the overall product plan is performed every 6 months. The prioritisation is performed in the form of a workshop where experts with different competences participate (see Table IV). The prioritisation is conducted using pair-wise comparisons and supported by the commercial tool Focal Point (Telelogic Focal Point - Software for Product Management and Product Portfolio Management 2009).

The components used as evaluation criteria are presented below following the order of their importance as defined by practitioners: (i) benefit (value) to company and alignment with product strategy; (ii) benefit (value) for potential customers; (iii) cost; (iv) how well the requirement improves non-functional characteristics.

For all three products the prioritisation and qualifying the criteria, such as requirements cost and value (for the organisation itself and potential customers), were mainly based on expert opinion (knowledge). Occasionally for estimating cost of a requirement additional material such as technical reports were used. When it came to estimating value it was completely based on expert opinion. The practitioners in general found it difficult to explain the mechanism behind estimation of requirements value. One of the product managers commented '*we estimate requirements value based on the knowledge of our product and our customers' needs*'.

4.2.2.2. Finding 4: Balance between Different Requirements Types at Prioritisation. At this stage of the practiced process all products considered both commercial and technical requirements. When it comes to the question of balance between the commercial and technical requirements the interviewed system experts expressed their concern that system and architectural improvements were receiving less attention compared to commercial requirements from e.g. market units and key-customers. '*Internal requirements connected to architectural and system improvement issues have a hard time competing with commercial features and often end up lower in the prioritised list of requirements*' mentioned one of the system experts. Another expert noted: '*system requirements receive attention when we can show that they are necessary for implementation of some important commercial requirement*.' System experts anticipated that this development could in the long run cause deterioration of the system architecture. The interviews system experts also expressed

DOI: 10.1002/spip

that describing the value of system requirements was not easy and that they needed guidelines for formulating the 'business case' for system requirements.

4.2.3. Results for release scope analysis and refinement (step 3 in Figure 2)

Analysis of the interviews with involved roles in this step, and data collected via observations of Release Scope refinement Board and Release Planning Board meetings (see Table IV) revealed that involved parties made a real effort to follow the agile way of working described in the process documentation. However, following lean principles to 100% when planning the contents of the future release proved to be problematic.

4.2.3.1. Finding 5: In Practice It Was Not Realistic to Follow Lean Development Principles Fully. The study results indicated that following the practices of evolving release scope (AP3) as described in the official documentation for Streamline (see Section 4.1.3) made it difficult to know which of the requirements included in the priority directive would be included in the final release. This created disturbances for product managers who were in need to have a good overview of release plan and its contents. As one product managers explained '*if we follow agile then I can not tell market units and influential customers what functionality will be delivered in the next release*.'

According to the interviewed line managers and product managers it was usual that the number of requirements included in the priority directive was more than the capacity of the organisation, thus not all requirements could be implemented. Consequently the product managers required confirmation from the R&D department on how many of the requirements in the priority directive would be possible to include in the release. This in turn meant that all requirements in the priority directive had to be analyzed to produce a reliable cost overview of the requirements included in the priority directive (according to interviewed line managers the available information on the cost from the previous pre-project steps was too high level to be trusted). This situation created problems since analyzing the entire scope of the release was against the lean philosophy of Streamline and line and project managers at the R&D department were afraid to fall back into the waterfall way of working.

In order to find a middle ground between lean principles and the needs of product management a commitment model was suggested according to which the R&D organisation would commit up to 50% of requirements in the priority directive from the start. This meant that the scope of the release would be defined ahead of development projects and it would contain the first half of all prioritised requirements. The remaining requirements would remain as candidates for in-scoping and would be considered only after the analysis of the committed scope was finished and necessary resources for producing the requirements assigned to the scope were secured.

4.2.3.2. Finding 6: Related topic: Different Attitudes Towards Agile in R&D and Product Management. Interviews revealed a difference between the attitudes towards agile principles in the R&D organisation and in the product management organisation. Line and project managers who are representing R&D were in general positive and enthusiastic towards agile, while product managers in general expressed concern about agile. As one product manager said '*I must feel confident about the contents of the release in order to be able to market our product and handle competition*'. It is interesting to note that the R&D organisation perceived the product managers concern about agile

DOI: 10.1002/spip

principles to be due to '*the old way of thinking*'. As some of the project managers said '*Product managers just have to find different ways to sell our products*'.

4.2.4. Summary

In this section the findings identified through interviews and observations will be analyzed with regard to the extent the practiced process follows the official Streamline process described in Section 4.1. After this, the findings from the interviews and observations are organised in practiced process symptoms, providing ground to discuss the implication of agile properties on MDPD needs.

Comparison between official and practiced processes: Finding 1 and Finding 3 show that for the steps of pre-screening and prioritisation the practiced process closely followed the steps outlined by the official process (see Section 4.1). Some differences were identified showing misalignment when it comes to the focus of the activities (for example in pre-screening the activities were less focused on value analysis than expected). In addition, the produced decision support material was not always written down as official reports on technical and business analysis (as mandated by the formal process), but often consisted of informal decisions based on expert opinion.

According to Finding 5 the largest difference between practiced and official process was identified in the step of release planning and refinement, where the practice of evolving release scope as prescribed by Streamline was in reality difficult to follow. It is important to notice that the step of release scope analysis and refinement represents the actual change from the waterfall approach to the new agile way of working with the requirements. According to the existing research on the extent of methodology adoption (Fitzgerald 1998) it is not uncommon that the prescribed development methodology in reality is not followed rigorously. However, in this case the interviews and observations provided clear indications that due to the problems associated with the old way of working, the practitioners have made every effort to follow the prescribed process. It was due to the misalignment with the needs of decision makers that the adaptations to the prescribed way of release content analysis and planning was necessary.

4.2.4.1. Practiced Process Symptom 1: Pre-project Activities are Focused on Commercial Requirements. Analysis of the findings for the pre-screening and prioritisation steps shows clearly that these initial pre-project activities are mainly driven by commercial requirements (see Finding 2 and Finding 4). Additionally according to Finding 4, commercial features are often prioritised over other types of requirements like architectural improvements.

4.2.4.2. Practiced Process Symptom 2: Evolving Release Scope Was Found Wanting in Relation to the Needs of Decision Makers. Finding 5 indicates that applying the concept of evolving release scope (AP3) was difficult for release planning purposes in a market-driven context. Thus there was a need to find a solution where agile properties would be adjusted to the needs of pre-project decisions. The introduced approach of a commitment model is an example of finding a middle ground between agile principles and the company's needs. Further, Finding 6 indicates a lack of understanding of the challenges and business constraints of pre-project decisions in a market-driven context.

4.2.4.3. Practiced Process Symptom 3: Requirements Value is Defined through Expert Judgement. Findings 1 and 3 reveal that practitioners do not have a clear mechanism for describing, evaluating

DOI: 10.1002/spip

and comparing value of different requirement types. According to Finding 3 evaluation of requirements value is mainly a tacit process based on expect opinion, knowledge of product's business case and customer needs. Moreover, data presented in Finding 4 indicate a need to define clear procedures and guidelines that can be used for representing different requirement types, for example system requirements.

5. DISCUSSION

In Table VI the symptoms of the practiced process identified through the case study are connected to the MDPD needs and agile properties that were presented in Section 2.3.

As shown in the table, *Symptom 1* and *Symptom 3* of the practiced process is associated with the agile properties AP1 and AP2 and is considered to impact the need to find an appropriate balance between different requirement types, such as commercial requirements, requirements connected to technology innovation and architectural and system improvements (N1 and N2). *Symptom 2* is connected to AP3 and is considered to impact the need to effectively plan the content of the future release (N3). The following sections elaborate on the implications of applying agile properties on pre-project activities in the MDPD context, and discuss possible solutions to decrease the gap between agile properties and needs in MDPD.

Table VI. Alignment of study results with agile and MDPD properties.

Agile property	MDPD need	Observed symptom	Possible impact
AP1. Feature orientation	**N1.** Balancing different requirements types	**Symptom 1:** Pre-project activities are focused on commercial requirements	– Short-term thinking
AP2. Reactive development	**N2**. Trade-off between market-pull and technology-push	**Symptom 3:** Requirements value is defined through expert judgement	– Possible architecture deterioration – Product integration problems – Limited ability for pre-emptive release of features and thus influencing markets
AP3. Evolving release scope	**N3**. Release content planning	**Symptom 2**: Evolving release scope was found unfitting for the needs of decision makers	– Limited possibility to influence and plan release content and related development cost – Limited understanding of total release value – Change management becomes difficult (no baseline to compare)

DOI: 10.1002/spip

5.1. Feature orientation vs. balancing different requirement types

In the MDPD context balancing commercial, architectural and quality requirements are extremely important (Ebert 2005, Regnell *et al.*, 2005). The system architecture is seen as one of the assets on which future releases will be built and maintainability issues are connected to the cost of supporting different versions of the products in the market over time (Tomaszewski *et al.*, 2008). The problems of finding a balance between features and architecture improvements have been acknowledged and reported by other researchers (Wohlin and Aurum 2005) and are common to Market Driven Requirements Engineering (MDRE).

The results of the conducted case study have shown the dominance of commercial requirements in Streamline, threatening to create an imbalance between features and other types of requirements not driven by the perceived immediate needs to the customer or market.

In the case studied in this article we can not claim that a perfect balance existed prior to the introduction of Streamline. However there is a risk that due to the agile properties of feature orientation and reactive development (AP1 and AP2), finding the balance might be even more difficult. This assumption partly is motivated by a lack of support for handling non-functional requirements in agile methods (Paetsch *et al.*, 2003, Sillitti *et al.*, 2005), as well as existing research where feature orientation and reactive development in agile methods is named as a threat for long-term goals such as maintaining a clean system architecture and other quality aspects (Boehm 2002, Tomaszewski *et al.*, 2008).

The risk of increased dominance of commercial requirements when applying agile methods can be explained by examining the definition of requirements value and requirements priority in agile methods.

As discussed in Section 2.3, agile methods focus on delivering rapid value for a customer, where requirements value and consequently its priority is strongly associated to the customer value of a requirement (Beck and Fowler 2001, Poppendieck and Poppendieck 2003, Williams and Cockburn 2003, Leffingwell 2007). This definition of a requirement's value in agile methods in combination with unclear procedures for defining requirements value (Practiced Symptom 3) can easily encourage prioritisation of commercial requirements which have a direct link to existing customers and overlook the value of requirements which cannot be easily linked to the customer, for example system and architecture related requirements.

The latest studies of requirements prioritisation practices applied in industry have revealed that having unclear procedures when it comes to definition of requirements value is not uncommon (Barney *et al.*, 2006, Lehtola and Kauppinen 2006). Thus for companies aiming to combine market-driven development with agile practices and still maintain a reasonable balance between the commercial and other types of requirements it is extremely important to (i) establish a clear picture of what is /should be perceived as valuable for a company, as opposed to value based only on the current needs of customers and how commercial and other type of requirements fit in this picture; and (ii) define clear procedures and guidelines for expressing the value of non-commercial requirements.

5.2. Evolving release scope vs. release content planning

Previously in this article it was argued that applying the concept of evolving release scope as defined in AP3 fundamentally limits product managers' ability to preemptively plan and control the contents

DOI: 10.1002/spip

and value of a release. The potential limitations of applying the evolving release scope concept in MDPD context can be explained by the assumptions of agile methods regarding development and business context in which a company operates (Section 2.3), as well as the simplistic view of agile methods when it comes to deciding the priority order of requirements and managing the release scope.

In agile methods all development is orchestrated by the priority order of requirements, making the priority list central and very important. The question is then how this priority is decided. Agile methods provide quite a simple solution to this where the task is left to the customer who decides priority based on the business value of a requirement. In order to help the customer prioritise, the developers provide some rough cost estimates, but mainly the order is decided by the business value (Beck and Fowler 2001, Poppendieck and Poppendieck 2003, Williams and Cockburn 2003, Leffingwell 2007).

In MDPD the priority and implementation order of a requirement is a function of the expected value and cost of a requirement, technical dependencies of a requirement and value dependencies between the requirements (Carlshamre 2002, Greer and Ruhe 2004, Saliu and Ruhe 2005, Ngo-The and Ruhe 2008). The mentioned difference in definition of priority implies that for the same set of initial requirements agile and MDPD will produce different priority orderings of these. More importantly, the definition of requirements priority in agile, in combination with the concept of the evolving release scope affects the possibilities to plan and follow up the contents and expected ROI of a release. This point is further on exemplified in Figure 3.

Figure 3 shows typical process steps of release content planning and evolution in an agile (to the left) and market-driven (to the right) context respectively and inputs and outputs for each step. Looking at the figure it is easy to notice that both scenarios follow the same process steps. In both of the scenarios there are collections of prioritised requirements that represent the initial release scope and both of the scenarios are iterative, where the development activities follow the priority order of requirements placed in the release scope. After each iteration there is a process step allowing analysis of delivered results and progress, as well as handling of changes, for example dealing with newly elicited requirements (*Req. x in the figure*). This analysis may result in re-prioritisation and an updated release scope.

A closer look at Figure 3 however uncovers critical differences between the two scenarios, most of which is caused by the difference in how the priority and implementation order of a requirement is defined in agile vs. MDPD.

The differences between the scenarios are summarised below: (i) priority order of requirements in the initial scope is different between the scenarios, resulting in differences in development activities. For example in the agile scenario the first iteration focuses on *Req. a*, and in the MDPD scenario on *Req. c*; (ii) requirements analysis and re-prioritisation step results in different content and requirements priority order in the updated scope of each scenario; where for example the newly arrived requirement *Req. x* in the agile scenario receives high priority and in the market-driven scenario does not. (iii) And finally in MDPD scenario pre-project decisions are tightly connected with defining and following up the total value and expected ROI of the final release (*in the figure this is indicated by assigning arbitrary values for total release ROI and total release value*) whereas in agile scenario this kind of analysis is difficult (*denoted by having question marks for the same variables*).

The definition of the priority order of the requirements in MDPD demands a proper understanding of all requirements which are assigned to the scope. This way it provides a base for estimating the

DOI: 10.1002/spip

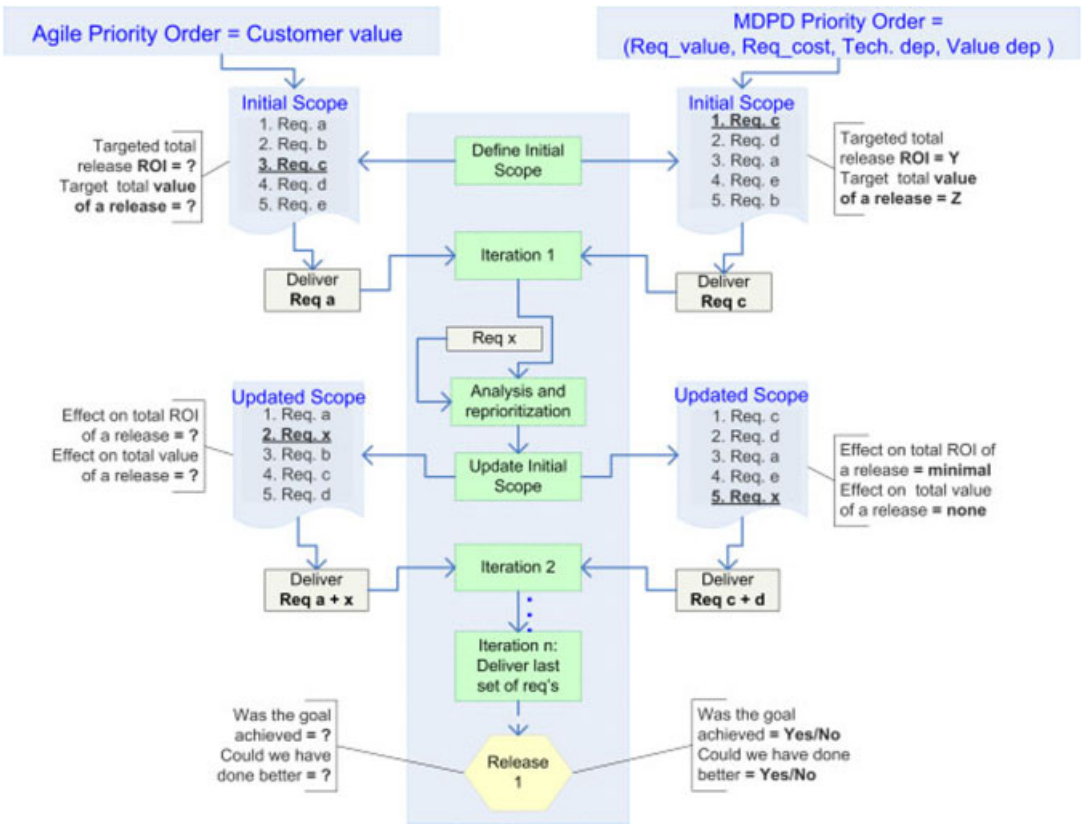

Figure 3. Differences in approach towards release planning between Agile and MDPD.

target ROI and total value provided by the requirements selection (Saliu and Ruhe 2005, Ngo-The and Ruhe 2008). This base is then used in order to plan, control, and follow up how well the selected list of requirement satisfies target business goals of a release. In addition it is important to gauge the consequences of how removing or adding a new requirement to a release scope affects the total value and expected ROI of a release, making it possible to deal with a change in an appropriate way.

The simple definition of the priority order of the requirements in agile methods combined with the property of an evolving release scope (AP3) does not allow an in-depth analysis of all candidate requirements (Beck and Fowler 2001, Poppendieck and Poppendieck 2003, Williams and Cockburn 2003, Leffingwell 2007). This makes it difficult to define the target ROI and total value to be delivered by the release and creates a risk of abusing the practice of 'delaying the decisions' where the decision on what will be included in the release is taken after the fact, and thus the target ROI and the total value provided by the release is not clear until after the delivery.

Results of the conducted case study have provided two important insights regarding this issue. Firstly the results uncovered difficulties to follow the practice of evolving release scope, mainly due

DOI: 10.1002/spip

to posed limitations to product management operations (see Finding 5). This result relates well to the MDPD characteristics presented in Section 2.2, and provides support for above-outlined discussion on misalignment between product management needs and limited possibility of managing and controlling release scope in agile methods.

Secondly the results have indicated different attributes towards agile methods in the organisation, showing that while R&D part of the organisation was in general positive to application of agile practices in development project, there was limited understanding of how agile properties affected product management (see Finding 6).

The above mentioned findings are important since they show that while agile is appreciated in separate development projects (Karlstrom and Runeson 2005, Svensson and Host 2005, Petersen and Wohlin 2008) they are not directly suited or designed for long-term product planning and product management, thus there is a need to find a way to apply agile methods in development projects without disrupting product management operations which are in need of more long-term planning and control.

The need for finding a compromise between agile and plan-driven approaches and tailoring agile methods to meet specifics of different product development and business contexts has been suggested by other researchers as well (Boehm 2002, Abrahamsson *et al.*, 2003, Cohen *et al.*, 2004, Fitzgerald *et al.*, 2006), however currently a concrete suggestion on how to accomplish this in MDPD context has not been suggested. The commitment model described in Finding 5 represents one example of such a compromise. However, it is important to notice that in order to determine the fist selection of functionality committed to by R&D, somewhat reliable information on the relative value and the cost of all requirements assigned to the release scope should be available. This implies that some pre-study work is still necessary. The question is what is good-enough, and how to support informed decision making without investing too much effort or over analysing the requirements (Fricker *et al.*, 2007).

5.3. Validity

Issues connected to construct validity; internal validity and external validity describe commonly known validity threats associated with empirical studies (Yin 1994, Wohlin *et al.*, 2000).

Construct validity is concerned with establishing the correct operational measures in order to assure that the proper information will be collected. This includes decisions on how the information sources are selected. The usual way to handle this threat is to use multiple sources of information. The case study at Ericsson was designed to use multiple sources of information and multiple ways of collection from documentation analysis and interviews to observation sessions during meetings. All these sources allowed for data triangulation. As shown in Section 3.1, studied documents, interviews, and meetings were carefully selected to achieve a correct representation.

When conducting investigations that are based on interviews, the amount and form of the interviews are important. In this case study budget and time considerations as well as availability constraints allowed for a total of 14 interviews. The interviews were semi-structured. Notes and conclusions made during the interview sessions were also confirmed by sending them to the interview subjects, where additional questions for clarification were attached where relevant. In some cases an extra interview was arranged, however these extra interviews are not included in the total count of the interviews in the study.

DOI: 10.1002/spip

Internal validity threats are usually important when examining the causal relationships. The case study presented in this article is mainly focused of studying the characteristics of pre-project activities. It has an exploratory character thus the internal validity threats can be considered as minimal. In the discussion section study findings are used to explore how well the study findings support the anticipated misalignment between agile properties and MDPD needs as described in Section 2.3. For example the dominance of commercial requirements in pre-project activities (Symptom 1) is connected to the agile properties feature orientation (AP1) and reactive development (AP2) and difficulties with applying the concept of evolving release scope (Symptom 2) is associated to the agile property of evolving release scope (AP3).

External validity is concerned with generalisability issues and is a threat that is common to case study research. Even though the case study in this report is conducted in a concrete industrial setting, we hope that the study results should be generalisable beyond the studied case since the investigated pre-project activities are not specific only for Ericsson but are common for most companies operating in a market-driven context.

6. CONCLUSIONS

In this article we have investigated the applicability of agile methods for MDPD, focusing particularly on pre-project activities such as requirements pre-screening, prioritisation and release planning. The presented work is considered to add value since it is one of the first contributions that investigates the applicability of agile practices in this context, providing extensive analysis and comparison of agile practices and needs in MDPD, as well as results from a large empirical study at Ericsson. This area is especially relevant considering the current hype and popularity of agile methods, and the contents and conclusions of this article should be interesting for any software company operating in a market-driven context planning to adopt agile practices.

The initial analysis and comparison of agile properties and MDPD needs (see Section 2.3) indicated a misalignment between the two. This was further confirmed by the results of the case study, where the impact of agile properties was studied for three different products developed at Ericsson. The outcome of the case study has provided confirmation of the misalignment between agile property 'evolving release scope' and the needs for release planning in the market-driven context. Further, case study findings also provided indications that 'feature orientation' and 'reactive development' might be a threat when it comes to balancing commercial and other types of requirements, and achieving a tradeoff between market-pull and technology-push as these aspects are not central in agile development.

Looking at the origin of agile/lean development the misalignment can be partially explained. Agile methodology having bespoke software development origin was developed with customer-developer relationships in mind and has a project focus. Thus agile methods such as Scrum and XP are lacking support for an overall long-term product development focus, normally associated with software product management activities in companies like Ericsson.

The main conclusion drawn is that agile methods do not directly support the needs of pre-project activities in MDPD. Application of agile properties in an organisation operating in market-driven context places limitations on product management activities, and may have a detrimental effect on long-term product development.

DOI: 10.1002/spip

Research results presented in this article indicate that agile principles in their current form should not be forced on product management tasks since they are not fit for it. In MDPD product managers need to maintain long-term focus, differentiate between customer and enterprise value and plan and control the scope of a release. However, this does not mean that product management can not benefit from agile/lean principles at all. The ideas from Lean SD can be reused to find ways for producing good-enough decision material for product management, enabling informed decisions far beyond the scope of any individual project. For example companies should benefit from practices which will allow minimising time and effort spent on analysing pre-project decisions. These practices should also guarantee that minimising effort does not come at cost of jeopardising product managers' ability to take informed decisions or increased costs in later stages of product development. The point here is that while it is important to produce fast, it is also important to maintain the big picture and understand how the achievements of individual development projects benefit both short- and long-term business goals of the company developing the product.

6.1. Future work

Through the investigation and uncovering of the impact of agile practices on software product management activities the authors of this article hope to create a foundation for finding ways to incorporate agile projects in MDPD without compromising long-term product goals.

The initial steps towards solving this issue will be connected to resolving the uncovered misalignment points between agile principles and product management needs. The future research will evolve in two directions: (i) the first direction will focus on helping software companies maintain an appropriate balance between different types of requirements even when following agile development principles. In order to achieve this we plan to work on separation of customer value from enterprise value and finding approaches for defining and comparing value of commercial and architectural requirements. (ii) The second direction intends to find ways of applying lean development ideas in software product and release planning situations. The key here is in finding just-enough level of analysis of the entire release scope without requiring excessive details but also allowing for the presence of decision support material (e.g. in the form of good-enough requirements). In this way the authors hope to facilitate defining practices for Lean SD.

ACKNOWLEDGEMENTS

We would like to express our gratitude to Ericsson, an industrial partner of the BESQ (Blekinge Institute of Technology, Sweden; http://www.bth.se/besq) research centre, for making this study possible. In the search for a middle ground between agile and plan-driven models Ericsson has taken an important step to integrate agile methods into their product development. Through the experience of using these methods, Ericsson started adding to state-of-the-art by investing effort in finding new and improved ways of applying agile principles to large-scale product development. These efforts are a part of joint ongoing research projects. Last but not least we would like to thank all the skilled and dedicated professionals at Ericsson for seeing opportunities and not problems.

DOI: 10.1002/spip

REFERENCES

Abrahamsson P, Warsta J, Siponen M, Ronkainen J. 2003. New directions on agile methods: a comparative analysis. *25th International Conference on Software Engineering (ICSE'03)*. IEEE Press.

Barney S, Aurum A, Wohlin C. Quest for a Silver Bullet: Creating Software Product Value through Requirements Selection. 2006.32nd EUROMICRO Conference on Software Engineering and Advanced Applications (EUROMICRO'06), IEEE Computer Society. Cavtat, Croatia.

Beck K. 1999. *Extreme Programming Explained: Embrace Change*. Addison-Wesley: Massachusetts, Boston MA,

Beck K, Fowler M. 2001. *Planning Extreme Programming*, *The XP series*. Addison-Wesley: Boston, MA.

Boehm B. 2002. Get ready for agile methods, with care. *Computer* **35**(1): 64–69.

Carlshamre P. 2002. Release planning in market-driven software product development: provoking an understanding. *Requirements Engineering* **7**(3): 139–151.

Carlshamre P, Sandahl K, Lindvall M, Regnell B, Nattoch Dag J. 2001. An industrial survey of requirements interdependencies in software product release planning. Proceedings of the Fifth IEEE International Symposium on Requirements Engineering, Los Alamitos, CA, IEEE.

Clements P, Northrop L. 2002; Software product lines: practices and patterns. *The SEI Series in Software Engineering*. Addison-Wesley: Boston, MA; 608.

Cohen D, Lindvall M, Costa P. 2004. An introduction to agile methods. *Advances in Computers* **62**: 1–66.

Dagnino A, Smiley K, Srikanth H, Anton AI, Williams L. 2004. Experiences in applying agile software development practices in new product development. 8th IASTED International Conference on Software Engineering and Applications, Cambridge, MA.

Dahlstedt A, Karlsson L, Persson A, NattochDag J, Regnell B. 2003. Market-driven requirements engineering processes for software products - a report on current practices. International Workshop on COTS and Product Software RECOTS, held in conjunction with the 11th IEEE International Requirements Engineering Conference, Los Alamitos, CA, IEEE.

Dybå T, Dingsøyr T. 2008. Empirical studies of agile software development: a systematic review. *Information and Software Technology* **50**(9–10): 833–859.

Ebert C. Requirements BEFORE the requirements: understanding the upstream impacts. 13th IEEE International Conference on Requirements Engineering (RE'05), IEEE Paris, France. 2005.

Erickson J, Lyytinen K, Siau K. 2005. Agile modeling, agile software development, and extreme programming: the state of research. *Journal of Database Management* **16**(4): 88–100.

Fitzgerald B. 1998. An empirical investigation into the adoption of systems development methodologies. *Journal of Information & Management* **34**: 317–328.

Fitzgerald B, Hartnett G, Conboy K. 2006. Customizing agile methods to software practices at Intel Shannon. *European Journal of Information Systems* **15**: 200–213.

Fricker S, Gorschek T, Myllyperkiö P. 2007. Handshaking between software projects and stakeholders using implementation proposals. *Requirements Engineering: Foundation for Software Quality*, *(REFSQ'07)*. Springer Berlin/Heidelberg: Trondheim.

Gorschek T. 2006. Requirements engineering supporting technical product management. *School of Engineering-Department of Systems and Software Engineering*. Blekinge Institute of Technology: Ronneby. 342.

Gorschek T, Davis A. 2008. Requirements engineering: in search of the dependent variables. *Information and Software Technology* **50**: 67–75.

Gorschek T, Wohlin C. 2004. Packaging software process improvement issues-a method and a case study. *Software: Practice and Experience* **34**(14): 1311–1344.

Gorschek T, Wohlin C. 2006. Requirements abstraction model. *Requirements Engineering* **11**(1): 79–101.

Greer D, Ruhe G. 2004. Software release planning: an evolutionary and iterative approach. *Information and Software Technology* **46**(4):243–253.

Hanssen GK, Fægri TE. 2008. Process fusion: an industrial case study on agile software product line engineering. *Journal of Systems and Software* **81**(6): 843–854.

Highsmith J, Cockburn A. 2001. Agile software development: the business of innovation. *IEEE. Computer* **34**(9): 120–122.

Ilieva S, Ivanov P, Stefanova E. 2004. Analyses of an agile methodology implementation. *30th Euromicro Conference*. IEEE Computer Society Press:

Karlsson L, Dahlstedt A, Nattoch Dag J, Regnell B, Persson A. 2003. Challenges in market-driven requirements engineering – an industrial interview study. Proceedings of the Eighth International Workshop on Requirements Engineering Foundation for Software Quality (REFSQ'02), Essen, Germany, Universität Duisburg-Essen.

Karlstrom D, Runeson P. 2005. Combining agile methods with stage-gate project management. *IEEE. Software* **22**(3): 43–49.

Koch AS. 2005. *Agile Software Development Evaluating the Methods for Your Organization*. Artech House Publishers: London.

Layman L, Williams L, Cunningham L. 2004. Exploring extreme programming in context: an industrial case study. *Agile Development Conference*, Salt lake city, UT; published by IEEE Computer Society Press.

DOI: 10.1002/spip

Leffingwell D. 2007. *Scaling Software Agility*. Addison-Wesley: Boston, MA.
Lehtola L, Kauppinen M. 2006. Suitability of requirements prioritization methods for market-driven software product development. *Software Process: Improvement and Practice* **11**(1): 7–19.
Cunningham W, Manifesto for Agile Software Development. 2001. [cited 200–09-30]; Available from: http://www.agilemanifesto.org.
Mann C, Maurer F. 2005. A case study on the impact of scrum on overtime and customer satisfaction. *Agile Development Conference (ADC'05)*. IEEE Computer Society: Denver, CO, published by IEEE Computer Society Press.
Mannaro K, Melis M, Marchesi M. 2004. Empirical analysis on the satisfaction of IT employees comparing XP practices with other software development methodologies. *Extreme Programming and Agile Processes in Software Engineering. Lecture Notes in Computer Science*. Lecture Notes in Computer Science, Springer: Berlin.
Middleton P. 2001. Lean software development: two case studies. *Software Quality Journal* **9**(4): 241–252.
Ngo-The A, Ruhe G. 2008. A systematic approach for solving the wicked problem of software release planning. *Soft Computing – A Fusion of Foundations, Methodologies and Applications* **12**(1): 95–108.
Noor MA, Rabiser R, Grünbacher P. 2008. Agile product line planning: a collaborative approach and a case study. *Journal of Systems and Software* **81**(6): 868–882.
North American and European Enterprise Software and Services Survey. Business Technographics. Survey conducted by Forrester Research inc. Available at http://www.Forrester.com/ER/Research/Survey/Excerpt/0, 5449, 436,00.html. 2005.
Paetsch F, Eberlein A, Maurer F. 2003. Requirements engineering and agile software development. 12th IEEE international Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises, IEEE, Linz, Austria.
Petersen K, Wohlin C. 2008. Issues and advantages of using agile and incremental practices. The Eighth Conference on Software Engineering Research and Practice in Sweden (SERPS'08), Karlskrona, Sweden.
Pettersson F, Ivarsson M, Gorschek T, Ohman P. 2008. A practitioner's guide to light weight software process assessment and improvement planning. *Journal of Systems and Software* **81**(6): 972–995.
Poppendieck M, Poppendieck T. 2003. *Lean Software Development – An Agile Toolkit*. Addison-Wesley: Boston, MA.
Potts C. 1995. Invented requirements and imagined customers: requirements engineering for off-the-shelf software. Proceedings of the Second IEEE International Symposium on Requirements Engineering, Los Alamitos, IEEE.
Regnell B, Brinkkemper S. 2005. Market-Driven Requirements Engineering for Software Products. In *Engineering and Managing Software Requirements*, Wohlin C, Aurum A (eds). Springer: Heidelberg, Berlin 287–308.
Robson C. 2002. *Real World Research: A Resource for Social Scientists and Practitioner-Researchers*, 2dn edn. Blackwell Publishers: Oxford.
Saliu O, Ruhe G. 2005. Supporting software release planning decisions for evolving systems. 29th Annual IEEE/NASA Software Engineering Workshop (SEW'05), IEEE, Greenbelt, MD.
Schwaber K, Beedle M. 2001. *Agile Software Development with Scrum*. Prentice Hall: Upper Saddle River, NJ.
Sillitti A, Succi G. 2005. Requirements engineering for agile methods. In *Engineering and Managing Software Requirements*, Wohlin C, Aurum A (eds). Springer: New York, NY, 309–326.
Svensson H, Höst M. 2005. Introducing an agile process in a software maintenance and evoluition organization. 9th European Conference on Software Maintenance and Reengineering (CSMR 2005). Manchester, UK.
Telelogic Focal Point – Software for Product Management and Product Portfolio Management. 2009. [cited 2009–01-22]; Available from: http://www.telelogic.com/corp/products/focalpoint/index.cfm IBM Corporation.
Tian K, Cooper K. 2006. Agile and software product line methods: are they so different?. 1st International Workshop on Agile Product Line Engineering: collocated with the 10th International Software Product Line Conference (SPLC), Baltimore, MD.
Tomaszewski P, Berander P, Damm L-O. 2008. From traditional to streamline development - opportunities and challenges. *Software Process: Improvement and Practice* **13**(2): 195–212.
van de Weerd I, Brinkkemper S, Nieuwenhuis R, Versendaal J, Bijlsma L. 2006. Towards a reference framework for software product management. Proceedings of the 14th IEEE International Requirements Engineering Conference (RE'06), IEEE Computer Society, Minneapolis, MN.
Williams L, Cockburn A. 2003. Agile software development: it's about feedback and change. *Computer* **36**(6): 39–43.
Wohlin C, Aurum A. 2005. What is important when deciding to include a software requirement in a project or release?. International Symposium on Empirical Software Engineering, IEEE, Noosa Heads, Australia.
Wohlin C, Runeson P, Höst M, Ohlson MC, Regnell B, Wesslén A. 2000. *Experimentation in Software Engineering: An Introduction, The Kluwer International Series in Software Engineering*. Kluwer Academic: Boston, MA; 204.
Womack JP, Jones DT, Roos D. 1991. *The Machine That Changed the World: The Story of Lean Production*. Harper Perennial: New york.
Yin R. 1994. *Case Study Research*. SAGE Publications: Thousand Oaks, CA.

DOI: 10.1002/spip