Key Aspects of Software Release Planning in Industry

Markus Lindgren[†] Rikard Land[‡] Christer Norström[‡] Anders Wall[*]

[†]ABB Force Measurement
Västerås, Sweden
markus.lindgren@mdh.se

[‡]Mälardalen University
Dept. of Computer Science
Västerås, Sweden
christer.norstrom@mdh.se
rikard.land@mdh.se

[*]ABB Corporate Research
Västerås, Sweden
anders.wall@se.abb.com

Abstract

Software release planning is the process of deciding what to include in future release(s) *of a product. Basically the problem can be seen as a company-wide optimization problem involving many stakeholders where the goal is to maximize utilization of the often limited resources of a company and turn them into business benefit. Saliu and Ruhe have proposed a set of key aspects for release planning methods, of which only a subset have been validated in industry. In this paper we use the Saliu and Ruhe key aspects as a starting point for identifying key aspects of release planning. To do this we have performed a multiple case study involving 7 international industrial companies, all producers of software intensive products. Our contribution is 1) a more strict meaning of a release planning key aspect, 2) validation of some of the aspects proposed by Saliu and Ruhe, and 3) an extension of the key aspects. We also capture state-of-the-practice for release planning in industry.*

1. Introduction

Release planning is a company-wide optimization problem involving many stakeholders where the goal is to maximize utilization of the often limited resources, such as budget and developers, of a company and turn them into business benefit [17]. The release planning results in a decision of what to include in future *release(s)* of a product. In making this decision one needs to consider how to make a product profitable both in the short- and long-term. As input to release planning are a set of *needs* that, when realized into a product, provides some business/customer value. Normally the cost of implementing all of the proposed needs is larger than the budget allocated to a release, therefore a decision needs to be made of what to include in a release and what to postpone. Also, the *set of needs* are required to be prioritized in order to maximize business value of the needs included in a release. In addition, there are constraints that need to be considered during release planning [17]. For example, time-to-market, dependencies between different needs, required competencies, competitors' product offerings, new technology, market demand, and quality aspects, as is illustrated in Figure 1.

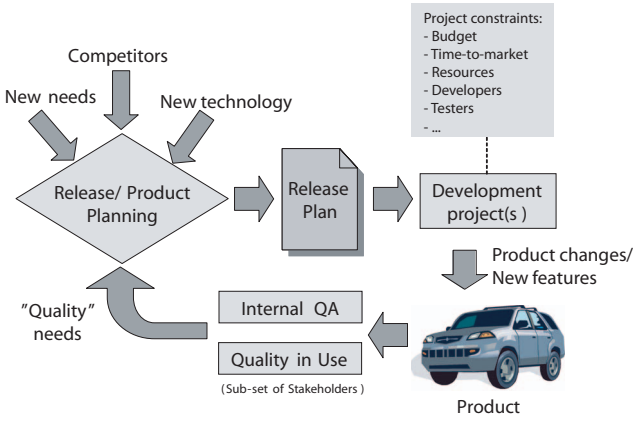

Figure 1. Overview of some relevant aspects of release planning.

A *need* is a proposal for a change that normally is negotiable to some extent; needs later become project requirements. However, there can also be parts of a need that are non-negotiable, i.e., constraints on the need. Examples of such constraints are legislation, time-to-market, and number of available resources. Usually there is a grey-zone in terms of what is negotiable and non-negotiable for a need.

The consequence of poor release planning can be, for example, increased technical debts [9] which are costly to address, increased need for replanning (resulting in project change requests), poor product quality, and lost business opportunities (when missing important market dates or when

1530-0803/08 $25.00 © 2008 IEEE
DOI 10.1109/ASWEC.2008.32

having "wrong" product features and/or quality). These issues often have impact on the profitability of a company.

In this paper we validate some of the release planning *key aspects* proposed by Saliu and Ruhe in [19], and identify additional aspects not covered by the aspects proposed by Saliu and Ruhe. To validate the importance of the proposed key aspects we have performed a multiple case study involving 7 industrial companies, most of them part of the Forbes Global 2000 list [8], where our main source of data is semi-structured interviews. Furthermore, as a part of this work we also more explicitly express what is meant by a release planning key aspect and, and we capture state-of-the-practice for release planning in industry. Our interviews have mainly been with *product management*, since product management of a company normally is responsible for performing release planning. The products developed by the companies in our study, contains *software*, *hardware*, and *mechanics* to different degrees; typically as part of an embedded system. However, our focus is still on release planning of the software part of these products.

One motivation for this work is that by studying release planning we can gain a better understanding for how investments in feature growth and quality improvements should be balanced in order to maximize business benefit in the long run, which is especially important for long-lived software intensive systems. To balance this successfully one must, e.g., be skilled in: requirements prioritization, "predicting" the market, and having flexible planning (in case market predictions are uncertain). Our hypothesis is that the companies that can handle the balance between investments in feature growth and quality improvements have a competitive edge over their competitors.

The outline of this paper is as follows: Section 2 presents the release planning key aspects proposed by Saliu and Ruhe, Section 3 presents a selection of related work, Section 4 presents details of our research method used in performing this study, Section 5 relates the proposed key aspects with collected interview data and qualitatively judges whether the proposed key aspects really are key aspects. Finally, Section 6 concludes the paper.

2. Reference Model

Saliu and Ruhe have proposed a set of *key aspects* for release planning methods in [19]. In lack of any more established set of key aspects we have used their model as an aid in comparing the release planning processes at the studied companies. It should be noted that these key aspects were proposed by Saliu and Ruhe in order to be able to compare different release planning *methods*, where some, but not all, of the aspects are supported by published work.

We use these key aspects for two different purposes, 1) confirm whether the proposed key aspects are *release planning key aspects*, and 2) for identifying key aspects not captured by the Saliu and Ruhe aspects. For sake of completeness we here briefly describe the Saliu and Ruhe key aspects of release planning methods (presented in [19]):

Scope Multiple releases need to be considered during planning, since some stakeholders are likely to become disappointed if their high prioritized requirements aren't included in the next release.

Time Horizon There are two ways of planning release intervals, 1) release planning with fixed and predetermined release intervals, and 2) release planning with flexible release intervals.

Objectives There must be some notion of an objective for release planning. "Typically it is a mixture of value, urgency, and risk, satisfaction/dissatisfaction, return on investment, etc."[19].

Stakeholder Involvement A stakeholder is any person or organization that has interest in a development project. Typical stakeholders are users, customers, developers, management, etc.

Prioritization Mechanism In typical organizations there is rarely enough resources to, all at once, develop all desired requirements. Therefore there needs to be some prioritization mechanism.

Technological Constraints Release planning needs to consider dependencies between requirements. An example of a dependency, requirement x cannot be implemented without requirement y.

Resource Constraints In development projects there are different resource constraints, e.g., release schedule, budget, and number resources such as programmers, testers, and designers.

System Constraints Usually organizations have an existing product offering for which new releases are being planned. The existing software architecture, code base, defect history etc. have impact on the effort, risk, and complexity trade-offs when adding new features.

Character and Quality of Solutions Offered From the description in [19] it is not really clear what this aspect is concerned with, we therefore quote the original description: "This dimension addresses the question of what is considered and accepted to be a solution of the problem. The range is quite large: A single solution versus a set of alternatives? An ad hoc solution that is not necessarily feasible versus a solution that is actually satisfying all (or most) of the constraints? Or a solution with undefined quality

versus a solution that achieves a predefined level of optimality?"[19]

Tool Support Release planning is a non-trivial problem involving lots of data, which can be simplified by the use of tools.

3. Related Work

Research within release planning is mainly focused on formalization of the release planning problem. This is typically done by formulating the problem as an optimization problem where *value*, e.g., customer value, should be maximized under a number of constraints; there are also tools implementing these algorithms, a comparison of these methods can be found in [19]. These methods are based on a number of variables to be estimated by experts that are used as input to these methods. In its most basic form *customer value* and *cost* are estimated [11], while other work consider more parameters [17]. In our study none of the companies use such methods/tools, indicating that either industry is unaware of these methods or that the methods don't provide the support required by industry. Part of our work is to find release planning aspects relevant for industry.

One exception to this line of research is Carlshamre [5] which argues that release planning is a "wicked problem", since release planning has characteristics matching those of wicked problems. Thereby making the release planning problem hard to properly formulate as an optimization problem. One of the characteristics of wicked problems are: *"There is no definitive formulation of a wicked problem. To define the problem is the same as defining the solution."*

Release planning has many similarities with *product planning* and *requirements prioritization*. One example of a method for requirements prioritization is presented in [12], which is based on the decision method AHP [18]. Several of the companies in the study use the tool Focal Point [7] as an aid during requirements prioritization, which is based on the method in [12]; we discuss this more in Section 5.5.

Wohlin and Aurum have investigated the importance of 13 different criteria used in deciding when to include a software requirement in a project or release [21]. They used brainstorming among researchers to derive their 13 critera and then created a questionnaire to be sent out to respondents. They conclude that business and management criteria are ranked higher than system/evolvability critera and that this is not an indication of this area being less important, rather that there is a need for better tools and methods for addressing these issues. In our work we have mainly used semi-structured interviews as data collection method, and hence have possibility of capturing additional criteria/aspects.

The Agile community [1] promotes close customer contact, regular planning meetings, and welcomes requirements changes, which is different compared to "traditional" software engineering approaches with more water-fall like or iterative development processes. In a sense, the exact contents of a release for an Agile project is decided during the project, which allows customers to change requirements late. This is simply a different development process; none of the companies in the study use this approach.

In earlier work we have looked into how software architects are involved in the release planning process in industry today [14], while this paper is focused on the release planning process. Both this paper and [14] are largely based on our collected interview data presented in [13].

4. Research Method

We have followed the recommendations by Yin [22] for multiple case studies.

In our study we have used semi-structured interviews as the primary data collection method, sometimes complemented by documents received from the interviewees. The main alternative, direct observations, has not been used due to the topic being studied containing company sensitive information and partly due to practical limitations.

We have addressed *construct validity* in multiple ways. First, there have always been two researchers present during each interview in order to reduce possibilities of misunderstandings. Second, interview notes have been taken during each interview, which have been sent to interviewees for approval. These notes have later been merged to a single description per company [13]. Third, we have had two test interviews to improve our interview setup. In total we have interviewed 16 people (excluding test interviews), typically 2–4 persons per company to achieve data triangulation.

To strengthen the *internal validity* of our study we have used multiple researchers when performing analysis and made use of pattern-matching techniques, and we have considered rival explanations.

To increase *reliability* of our study, all collected data, and derivations thereof, are stored in a database accessible only to the researchers in the study, e.g., interview notes and merged notes per company. In addition, the study design is documented, which includes the interview questions. Using this material it is possible to trace conclusions to collected data, and vice verse.

Thanks to industrial contacts we have been able to find a relatively large number of companies and persons willing to participate in our study, which aids in increasing the *external validity* of our results. Still, there is risk that the selection is not fully generalizable to other domains and/or nationalities/cultures.

To counter *researcher bias* we have been multiple researchers at most of the steps of this study. Furthermore, companies that the researchers in the study are affiliated with are excluded from the study.

Case	1	2	4	5	6	7
Volume	VH	H	VH	L	M	L-M
% Software	L-M	L-M	H	L-M	M	H
% Hardware	M	M	H	M	M	M
% Mech.	H	H	L	H	M	L
Employees	H	H	VH	L-M	VL	VL-L
% in R&D	VL	VL	H	L-M	VL	VL

Table 1. Charachteristics of companies in the study, excluding Case 3, where VH = Very high, H = High, M = Medium, L = Low, and VL = Very low.

For confidentiality reasons there are no company names, no names of interviewed people, and no absolute numbers on, e.g., budget and number of developers, in this material, but where possible we present relative figures.

4.1. Cases

Here we provide an overview of the seven companies in the study, a more complete description of the companies, and their release planning processes, can be found in [13]. Due to the size of these companies our study is limited to specific parts within these companies.

Table 1 presents some relative data concerning the characteristics of products developed, produced, and sold by the studied companies in order to provide a feel for their main characteristics. In Table 1 *Volume* refers to the produced product volume, while the rows *% Software*, *% Hardware*, and *% Mech.* is our subjective judgment of the products' software, hardware, and mechanical content, which in turn reflect the amount of resources these companies invest in these areas. Case 3 is excluded from the table since it is a consulting company that has no products.

5. Release Planning Key Aspects in Industry

In this section we relate data captured during our interviews with the key aspects proposed by Saliu and Ruhe [19] and we qualitatively determine the industrial relevance of these key aspects, i.e., judge whether these aspects really are key aspects of release planning in industry. We also identify additional aspects not covered in [19]. To be able to determine this we need an approximate definition of a release planning key aspect.

> *We consider an aspect to be a release planning key aspect if, and only if, the* consequences *of not considering the aspect can lead to additional costs that are unacceptable.*

The limit for "unacceptable costs" is clearly subjective and context-dependent. The remainder of this section has one sub-section per release planning aspect. For each release planning aspect we first describe *data* captured during our interviews, and then *discuss* and qualitatively determine if the aspect is a release planning key-aspect.

5.1. Time Horizon

Data: All of the companies in the study use a fixed release cycle; the release cycle varies from 6 to 18 months in the studied companies. Also, most of the companies in the study provide similar reasoning for the choice of the release cycle, basically it is a trade-off between time-to-market and organizational efficiency.

Most companies report a fixed release cycle being a good way of "pacing" large organizations. The following description captures the reasoning of many of the companies in the study quite well:

Case 6: The motivation for having 2 releases, instead of 5–6 as before, is primarily to improve internal efficiency for the company as a whole. Having 2 releases per year increases time-to-market, which can result in lower revenue and increased costs, since product improvements will later become in use. However, at the same time it will reduce the administrative overhead associated with a release, e.g., updating of price lists, update of manuals, educate the organization in the contents of the release, such as, local sales offices and the part of the company building application on top of the standard software.

Most of the interviewees also provide similar reasoning when there is trouble meeting a release date. The options basically are 1) change scope of the release by reducing its functionality, 2) post-pone the release date, or 3) if possible, redistribute resources within the organization (also globally). However, when faced with such a decision most companies prefer to reduce scope of the release to keep the release date. The following description is illustrative:

Case 7: It is always preferred to keep the target date for the release. The motivation for this is that it is more important to reach the market with the functionality that could be developed within 18 months, rather than letting a few projects delay the entire release. The project(s), or functionalities within projects, which are late may cause loss of a few orders, still it is more important to reach the market with the other product improvements such that the effect of the large amount of resources spent on the release becomes visible.

In addition to the main releases, the companies also have maintenance releases, for bug fixes and minor improvements, which not necessarily have a fixed release cycle. There can also be internal releases, as is the case for

Case 4, where there are internal releases every 6^{th} week going through system testing and then to the application part of the organization.

Discussion: Time horizon basically is a tradeoff between time-to-market and internal efficiency. Time-to-market controls when revenues can start growing for new features, and is sometimes a crucial factor for product success [4]. A company's internal efficiency controls how well the organization as a whole is able to develop, produce and release new features to the market. For example, Case 6 and Case 7 have increased their release cycle such that more time can be spent on development, in relation to overhead activities such as update of price lists, training in using the new release, and update of manuals. Clearly, the tradeoff between time-to-market and internal efficiency will be crucial to any company. We conclude *Time horizon* being a release planning key aspect.

5.2. Scope

Data: The release being described by most interviewees is the next release, few explicitly speak about the content of future releases. Partly an exception to this is Case 2 which during their integration test meetings discuss three releases: 1) the release decided during the previous meeting, 2) the current release, and 3) future releases. However, all companies in the study have product strategies and/or planned features, which are beyond the release being planned.

Perhaps more importantly, the scope of the current release needs to be managed continuously during the release project, since there may be unforeseen issues during a release project that needs to be dealt with. For example, there can be customer requests or potential customer orders that can have impact during a release project, such as, the need to add some feature required by a specific customer. There can also be technical difficulties that requires a release to partly be replanned.

Case 4 has an interesting way of dealing with this. One basic understanding concerning release planning at the company is that *there will be changes* during a release project, e.g., there will always be needs which aren't thought of during initial planning. To cope with this the initial release plan must not assign more than 50% of the release budget, the remaining 50% is planned to be used for needs and changes during the release project. Several other companies in the study assign 100% of the release project's budget during initial planning, which usually result in some change requests. Case 7 had one example where for one release about 30% of the initially planned requirements didn't make it to the final release. Furthermore, about 15% new requirements where added during the release project.

Discussion: When discussing the "scope" aspect with interviewees in the study the picture is not as clear as for Time horizon. First of all what most people intuitively think of when discussing "scope", is the feature content of a release. For example, if a project is running behind schedule its scope can be reduced, i.e., remove features from the current release, such that the target release date can be met. This implies that the choice of the word "scope", and the meaning suggested in [19], can lead to misunderstandings.

Second, in our interviews none of the interviewees discussed the importance of planning more than one release at a time; still one finding in [5] is that planning only one release at a time is not sufficient. However, what the interviewees discuss is if a need cannot be included in the current release, e.g., due to budget constraints, they postpone it to the next release. Later when it is time to plan the next release, the needs which didn't make it into the previous release usually receive higher priority. The point is that the investigated companies really don't *plan* more than one release at a time. However, there may be differences between different product domains and/or companies.

In terms of our interpretation of what a key aspect is, the consequences of not considering Scope (as described in [19]) is not sufficient to be classified as a key aspect. The basis for this judgment is a qualitative assessment of what the interviewees have discussed in relation to this.

5.3. Objectives

Data: Typically the primary objective of release planning at the studied companies is to obtain the best possible return-of-investment. None of the companies, with the exception of Case 3, has an explicit function describing the objectives as stated in [19], still there are objectives that the companies aim for. Case 1 has *attribute profiles* that describe desired properties for their products which is based on their company profile, for example, performance and safety can be part of such a profile (details in [13]). Case 2 has 8 *core values* and 3 *prestige values* that are prioritized higher. Case 3 proposes the use of the explicit function *measure of effectiveness* (MoE) defined in IEEE 1220 [10]. Case 7 has a release profile, with roots in the long-term product vision. Case 4, Case 5, and Case 6 generally start from a more general business value viewpoint, i.e., how to turn a need into business benefit.

Sometimes these objectives are related to a company and/or product strategy, and often there are objectives aimed at product *feature content*, product *quality*, and *cost-cut* (mainly related to production cost). Examples of other aspects that are relevant during release planning is sales trends for different products, number of sold product options (e.g., which options can be removed), and competitors product offerings [13]. For these companies there is often a lowest level of acceptable product quality and production cost which must be achieved before a product can be released.

Discussion: The Objectives as stated in [19] in a sense describe the desired properties for a product, or stated dif-

ferently, the goals of the product. Naturally it is desired to during release planning maximize the objectives, as is for example done in research such as [11, 17]. Case 1, Case 2, and Case 3 have a clear product strategy, i.e., their objectives are clearly defined and each release aims for optimizing these values. There are objectives for Case 4, Case 5, Case 6, and Case 7, but these are not as explicit as for the other cases. Without Objectives there is no "vision" of where the product should be aiming; without Objectives one more or less rely on chance/randomness. Objectives is a key aspect of release planning.

At the studied companies the objectives are often imprecisely defined and in lack of good data, people generally use "gut-feeling" to different degrees in making judgments concerning what to include in future releases.

5.4. Stakeholder Involvement

Data: It is clear that there are many different stakeholders that have interest in the contents of a release. However, there are relatively large differences between the companies in terms of which stakeholders 1) propose new needs, 2) are involved in making the release planning decisions, and 3) are involved in preparing decision material for the release planning decision. Below we discuss a few different examples of stakeholder involvement.

In Case 1 a relatively large focus is placed on capturing the needs of end-customers, which, for example, is performed using quarterly customer clinics and surveys; both by the company itself and by independent contractors. In terms of proposing new (internal) needs the company looks at three different aspects, which are *product features*, *quality*, and *cost-cut*. These areas are handled by three different parts of the organization. Before performing any product development, R&D investigates the consequences of making the product changes in pre-studies and returns the result of the pre-study to product management, allowing them to adjust prioritization of needs.

In Case 4 the investigated part of the company develops a product platform which is used by two other parts of the company, C_1 and C_2, when building products. Case 4 is focused on step wise refinement and step wise prioritization of the proposed needs. R&D is involved in pre-studies, feasibility studies, and, of course, project execution. When *collecting candidate needs* for the product platform they emphasize that each part must "speak with one-voice". What they mean is that before the needs from C_1 and C_2 reach product management for the platform they must internally prioritize their needs. In addition, the *system responsibles* from C_1, C_2, and for the product platform must, in the same way, speak with one-voice to product management for the product platform, as is illustrated in Figure 2.

In Case 5, Case 6, and Case 7 we see lower involvement of R&D and more focus on product management or sales

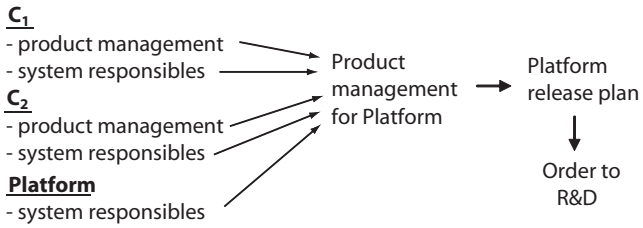

Figure 2. Collection of needs using "one-voice" from each organizational part.

representatives to propose new needs. Generally, in Case 4, Case 5, Case 6 and Case 7 there seems to be low customer involvement. More examples of stakeholder involvement in our study is available in [13].

Another kind of stakeholder is upper management, which usually decides the budget available to the development of a new release and budget for the different parts of the development organization. Hence, this further constrains the release planning problem, since this partly controls the available resources and the available competence. Usually upper-management can set some of the focus areas for a new release, which further constrains release planning. Hence, in a way, parts of the release planning decisions are made while deciding the budget for the next year.

Discussion: Stakeholder involvement can be seen from several different viewpoints. For example, an end-user may be interested to know if his/her "most sought after feature" will be included in the product, while a developer may be interested in the release plan itself being sound, e.g., is the assigned time and cost reasonable? In a way the Objectives (see Section 5.3) captures part of this aspect, since, if the Objectives are good the need for end-user involvement can be lower. It can also be seen as the end-users "validate" the objectives; if we are developing something the customer is not interested in, then we are simply developing the "wrong" product. This again is a motivation for Objectives being a key aspect of release planning. Via stakeholders involvement it is also possible to receive feedback on how the current product is operating, e.g., is there a need for quality improvements in an area? Without stakeholder involvement there is risk of developing the wrong product or the release plan being "unsound", therefore stakeholder involvement is a key aspect of release planning.

5.5. Prioritization Mechanism

Data: It is clear that the companies need to prioritize what to include in a release, since often, according to data collected in the study, the proposed set of needs often is required to be reduced by a factor of 3 to 4 in order to fit within the budget of the release project. At the studied companies prioritization is primarily handled by group

discussions. The starting point for the prioritization are the objectives discussed in Section 5.3.

Four of the studied companies report using the tool Focal Point [7] as an aid in prioritization. Focal Point is based on the Analytic Hierarchy Process (AHP) [18, 12] and uses pair-wise comparisons of cost and value. The interviewees report being most pleased with Focal Point's ability to structure discussions, gain better understanding of the needs, and to reach group consensus.

Discussion: Data collected in the study indicate that there often are 3–4 times as many proposals as can fit within the budget of a release project. This combined with Objectives and Resource constraints being key aspects of release planning results in there being a need for prioritization, where the goal is to maximize the benefits of the Objectives and minimize negative effects from the Resource constraints. This reasoning motivates the Prioritization mechanism being a key aspect of *release planning methods*, but not a key aspect for release planning. In our investigated cases it is uncommon to use a strict prioritization mechanism, or as one interviewee phrased it "it is not suitable to by mathematics compute what needs to be done".

Instead people generally resort to using "gut-feeling". This "gut-feeling" can be based on many different things such as: what is of benefit for the company, benefit to my own department, benefit to my own country (in case of distributed development), and benefit to my own career. There can be many such reasons that affect how people argue/reason during release planning. To increase the chances of their own proposals getting through, product management, and supposedly people generally, use lobbying, sell-in, and politics. Our study indicates that the release planning decisions are to various degrees affected by gut-feeling, lobbying, politics, and strong individuals. A few illustrative examples from the interviews are given below (more examples in [13]):

Case 1: "Decisions concerning how different aspects should be prioritized is primarily handled by argumentation". To get your own will through it is important to "lobby for your own case".

Case 4: "Strong individuals and people in strong positions more easily get their proposals through."

Case 6: "Release planning decisions are often handled by politics, trends, and opinions of upper management."

This is rather different compared to research, such as [11, 17], where an explicit function is optimized.

5.6. Resource Constraints

Data: The resources available to a project, whether it be money, equipment, designers, programmers, testers etc., have impact on the probability of meeting set project goals. In most of the cases in this study the R&D organization is responsible for reporting the number of available resources to product management. This is an input required for determining how many needs can fit within a new release.

Below are some examples from the different cases supporting that resource constraints are considered during release planning:

Case 4 Usually there are limited resources that are capable of performing pre-studies, since these normally require specific expertise. This resource constraint makes it necessary to start pre-studies at different points in time.

Case 4 To be able to perform a good release plan it is required that the R&D organization keeps track of their available engineering hours, and within the different competence areas these hours exist, and the time required for competence build-up within other areas. In practice product/release planning must consider competencies available within different areas.

Case 5 The total amount of available resources within the company, in form of development capacity, has impact on planning and marketing. For example, during periods with high order volume, fewer resources can be used for new development, which can possibly result in older components being used during order construction; often it requires fewer engineering hours to use existing components, rather than using new components which can really be a better choice.

Case 6 When working on the budget for the next annual year an engineer is treated as one engineer without considering the competence of individual engineers. The budget is set such that all engineers have 100% work load. Later, during project planning and execution some tasks need to be performed by engineers with specific competence, since all engineers aren't capable of performing all tasks, it turns out there aren't sufficiently many engineers available with the required competence. This is partly one of the causes of projects running behind schedule.

Case 7 Another complicating resource issue is that budgets for different development departments and release time are decided before the content of the next release has been decided. This makes it even harder to optimize for customer value in the release.

Discussion: One important consideration brought up by most of the interviewees is to keep track of the available R&D resources, as well as their competencies. Since without such knowledge it is not possible to determine how many features is suitable to include for development in a release. Simply, *without any resources there will not be*

any release. Resource constraints is clearly a key aspect of release planning, since without considering resource constraints the consequences can, e.g., be unrealistic release plans and/or under-utilization of existing resources; high R&D utilization is a strong focus area for Case 4 [13].

In our study there are additional constraints not mentioned in [19], which can be considered being resource constraints, that are relevant to consider during release planning. For example, there can be organizational aspects, as is the case for distributed development, similar considerations can also exist when sub-suppliers are used.

5.7. System Constraints

Data: We have earlier studied how system constraints are considered during release planning [14]. In short, system constraints are mainly considered via pre-studies, which result in decision material. The decision material sometimes results in product management adjusting the proposed needs' priorities.

It should be noted that the studied companies are rather different in how they consider system constraints during release planning. Some of the companies use pre-studies (performed by R&D) to refine the proposed needs and for investigating the consequences of those needs on, e.g., the existing system. However, some of the companies use less "structured" pre-studies, which have lower possibility relevant aspects of the existing system [13].

Discussion: Today most product development is required to be based on existing systems for economical and time-to-market reasons [15]. This forces existing systems to be evolved by adding features or improving existing features. Furthermore, this implies that in order to build something new one must have good knowledge of what can be reused from existing systems, such that development costs can be minimized. In our study the constraints imposed by existing systems is determined by using pre-studies to investigate the consequences of proposed needs, as is done in Case 1, Case 3, Case 4 and possibly Case 2. This aspect seems to be weaker in Case 5, Case 6, and Case 7.

The purpose of the pre-studies is to refine the proposed needs, increase the accuracy of cost and time estimations, and estimate consequences on the existing system. The consequence of not considering the existing system's constraints can result in cost and time estimations being higher than can be tolerated (partly based on the conclusions in [15]). Hence, System constraints is a release planning key aspect.

5.8. Technical Constraints

Data: In our study we have observed that for embedded systems it can be the case that software release planning has to adapt to planning made for electronics and/or mechanics. Since normally electronics and mechanics follow a more water-fall like development cycle, while for software there exist many iterative and Agile approaches to development [1]. For electronics and mechanics there can be production times or other aspects that hinder them from being delivered faster and/or iteratively.

A few examples from the cases are presented below:

Case 2: As far as possible they try to keep projects apart such that there are no dependencies between projects, since there is a higher risk of failing to meet set target dates in case there are project dependencies.

Case 4: The company is large and there are strong dependencies between different parts of the company. For example, the platform is required by A and B, and if a release is delivered late, or with poor quality, it can have significant impact on the success of A and B. As one interviewee stated it, "quality must be there and it must be delivered on time".

Case 6: A difficulty is in handling the complexity of the product, there are often many projects running in parallel with dependencies between them, and they can be of different nature, e.g., hardware and software.

Discussion: One issue that was brought up by all companies in the study is the problem with late detection of project dependencies. The problem is not caused by lack of methods for project follow up, but rather it is a problem of scale. As one interviewee said, when referring to follow up of projects and needs using an Excel sheet, "it is like looking through a straw". A release project within most of these companies consists of $20 - 30$ sub-projects, which in total can be working on more than 150 needs. These needs do have dependencies of various types, e.g., dependencies between hardware and software. Dependencies that are detected in time are normally no problem, since then it is possible to make plans that minimize/eliminate these problems.

Lately detected dependencies is a problem that degrades the efficiency of development and can lead to project delays. These consequences motivate *Technical Constraints* being a key aspect of release planning. The problem is to early identify all dependencies. Furthermore, for several of the companies in the study software planning is subordinate to planning of electronics and mechanics.

As a related example, Carlshamre et. al. [6] has performed a survey where it has been discovered that only 20% of the requirements are singular, i.e., independant, while the remaining requirements do have interdependencies to other requirements. Furthermore, 20% of the requirements stand for 75% of the interdependencies. Thereby indicating the scale of the problem in industry.

5.9. Character and Quality of Solutions Offered

Data: We basically have no data collected during our interviews which can be directly related to this, besides that the interviewees consider different release plans.

Discussion: As "Character and Quality of Solutions Offered" is described and used in [19] it seems as if it describes *properties* of the solution, or sets of solutions, produced by a tool/method. For example, do we get the top five solutions, a single solution, or a solution (or solutions) that come closest to meeting set constraints (in the case when all constraints cannot be fulfilled)? Hence, this description is related to properties of a solution of a release planning method/tool and consequently can not be seen as a key aspect for release planning (but certainly for a release planning method).

5.10. Tool Support

Data: All companies report using tools to aid in their release planning activities, but their uses of the tools differ. Examples of tools used are Focal Point, Roadmapper, Excel, and various in-house developed tools. As an example, in Section 5.5 we briefly discussed the benefits of using the tool Focal Point.

The uses of the tools differ between the companies. For example, some use the tools as a place for storing proposed needs, as an aid in prioritization, and for ordering development projects.

Discussion: It is clear that release planning is a non-trivial task, knowledge intensive, involves many different stakeholders, and many uncertain factors. Hence, if a tool can help in improving the efficiency and/or reduce probability of mistakes being made, this will for certain be appreciated by product managers. However, even if a tool helps in performing release planning, doesn't mean that it makes it a key aspect of release planning. The use of Tools as a "key aspect" in [19] is to enable comparisons of different release planning methods, and there it makes sense to compare the tool support for different methods. According to our critera, tool support is not a key aspect of release planning.

5.11. Short- and Long-Term Planning

In this section we discuss an aspect not covered by the key aspects proposed in [19]. Within the software engineering community it is known that as a software system ages the cost of change will increase [20, 16]. It is also known that aging is inevitable, just as it is for humans. However, there are methods that can be used to reduce the costs of aging, which is one aspect addressed by the Agile community where focus is placed on testing and continuous refactoring [1, 3, 9] in order to maintain good design and structure. *Technical debt* refers to "software aging" costs that are not attended to [9], which hence need to repaid at a later time. Parnas says that "our experience with software aging tells us that we should be looking far beyond the first release to the time when the product is old" [16], indicating there being a need for considering long-term aspects.

Case Interviewee	1 A	1 B	2 C	2 D	3 E	4 F	4 G	5 H	5 I	6 J	6 K	6 L	6 M	7 N	7 O	7 P
SW Edu.	M	N	L	N	-	N	L	N	N	M	N	M	?	M	H	N
SW Exp.	Y	N	N	N	-	N	N	N	N	Y	N	Y	?	M	N	N
RP Role	N	Y	N	N	-	Y	N	Y	Y	Y	Y	N	N	Y	Y	Y
Awareness	M	L	L	L	-	M	L	L	L	L	L	M	-	M	L	L

Table 2. Architectural awareness of interviewees A–P, where H=High, M=Medium, L=Low, Y=Yes, and N=No.

In our study we have found there being little attention to long-term planning for software aspects, such as, will there be a need for an architectural change to accommodate new changes [2]? Basically the companies in the study can be divided into those that perform "structured" pre-studies, where there is possibility of bringing up these issues, and those that do not perform structured pre-studies [14]. In this study Case 1, Case 3, Case 4, and possibly Case 2 use structured pre-studies, while Case 5, Case 6, and Case 7 have little R&D involvement thereby also having low possibilities of addressing these issues. This conclusion is in line with the results of Wohlin and Aurum [21].

We have discussed with the interviewees how they reason when there is a need to chose between adding a feature and performing an architectural improvement. Our conclusion is that there, generally, is low awareness of these issues among product management. One possible reason for this is that product management today, in the studied product domain, seem not to have a background in software engineering, as is shown in Table 2. In Table 2 there are columns for software development education (*SW Edu.*), software development experience (*SW Exp.*), whether the interviewee takes part in release planning (*RP Role*), and finally, our qualitative conclusion concerning the architectural awareness for each interviewee; the value *low* in the software education column means that the interviewee has taken a basic course in programming. (The gray table cells in Table 2 mark interviewees where we have had to resort to more detailed analysis to come to our conclusion.)

As a result of this there seems to be few long-term considerations in release planning in industry, which is related to the problem of balancing investments in feature growth with quality improvements as we have reported in [14]. If only short-term considerations are made in a release there is considerable risk quality problems in later stages, e.g., increased technical debts [9, 16] and/or problems poor customer/stakeholder satisfaction. Deferring these kind of

"debts" often has an associated interest rate, i.e., the longer one waits to repay the debt, the higher its cost will be. We consider this being a key aspect of release planning.

6. Conclusion

We have performed a multiple case study involving seven different industrial companies and investigated their release planning processes, with focus on the evolutionary phase of development. Saliu and Ruhe have proposed a set of key aspects that need to be considered by a release planning method, but some of these aspects have not been validated in industry. We have used their key aspects as a starting point for identifying key aspects for release planning.

The contributions of this paper are: 1) a more explicit formulation of what is meant by a release planning key aspect, 2) findings related to the key aspects proposed by Saliu and Ruhe, 3) a description of state-of-the-practice for release planning in industry, and 4) identification of a new key aspect for release planning (not part of the Saliu and Ruhe key aspects). The key aspects of release planning are: *Objectives*, *Resource constraints*, *Technology constraints*, *System constraints*, *Time horizon*, *Stakeholder involvement*, and *Short- and Long-term Planning*, where our contribution is the latter key aspect. If only short-term considerations are made in a release there is considerable risk of quality problems in later stages, e.g., increased technical debts and/or problems with poor customer/stakeholder satisfaction. Deferring these costs can result in even higher costs at later stages, it is therefore key to be able to balance investments in feature growth with quality improvements. Today too little attention is placed on this problem.

7. Future Work

Based on this paper it is hard to give practical advice to practioners on how to improve their work practices. In essence this work is a starting point for future work aiming at developing useful guidelines for release planning. In this section we propose four directions for future work. First, would be to validate the conclusions made in this paper by a quantitative survey; our study is qualitative. Second, it would be interesting with a more detailed study of the effect of short- and long-term planning for companies developing products with different life-cycle lengths, e.g., compare companies producing products with a life-cycle of 2 years with companies producing products with a life-cycle of 15–20 years. Third, methods and/or guidelines should be developed that aid in improving the balance between short- and long-term planning. One fourth direction would be to look into how different project models impact release planning, e.g., differences for organizations using Agile methods and Rational Unified Process. In our future work we aim at addressing the third and possibly the fourth point.

References

[1] What Is Agile Software Development? Web-site: http://www.agilealliance.org, 2007.

[2] L. Bass, P. Clements, and R. Kazman. *Software Architecture in Practice* 2^{nd} *edition*. Addison-Wesley, 2003.

[3] K. Beck. *Extreme Programming Explained — Embrace Change*. Addison-Wesley Professional, 1999.

[4] R. Calantone and C. Benedetto. Performance and time to market: accelerating cycle time with overlapping stages. *IEEE Transactions on Engineering Management*, 47, 2000.

[5] P. Carlshamre. Release Planning in Market-Driven Software Product Development: Provoking an Understanding. *Requirements Engineering*, 7(3):139–151, 2004.

[6] P. Carlshamre, K. Sandahl, M. Lindvall, B. Regnell, and J. N. och Dag. An Industrial Survey of Requirements Interdependencies in Software Product Release Planning. In 5^{th} *IEEE International Symposium on Requirements Engineering*, pages 84–91. IEEE Computer Society, 2001.

[7] Telelogic Focal Point. Web-site: http://www.telelogic.com, March 2007.

[8] Forbes Global 2000. Web-site: http://www.forbes.com, March 2007.

[9] M. Fowler, K. Beck, J. Brant, W. Opdyke, and D. Roberts. *Refactoring: Improving the Design of Existing Code*. Addison-Wesley Professional, 1999.

[10] IEEE Std 1220-2005, IEEE Standard for Application and Management of the Systems Engineering Process, 2005.

[11] H.-W. Jung. Optimizing Value and Cost in Requirements Analysis. *IEEE Software*, 15(4):74–78, 1998.

[12] J. Karlsson and K. Ryan. A Cost-Value Approach for Prioritizing Requirements. *IEEE Software*, 14(5), 1997.

[13] M. Lindgren. Release Planning in Industry: Interview Data. Technical Report MDH-MRTC-219/2007-1-SE, Mälardalen Real-Time Research Centre (MRTC), 2007.

[14] M. Lindgren, C. Norström, A. Wall, and R. Land. Importance of Software Architecture during Release Planning. In *WICSA 2008*. IEEE Computer Society, 2008.

[15] Mustapic, Wall, Norström, Crnkovic, Sandström, Fröberg, and Andersson. Real World Influences on Software Architecture. In *WICSA 2004*. IEEE Computer Society, 2004.

[16] D. Parnas. Software aging. In 16^{th} *International Conference on Software Engineering*. IEEE Computer Society, 1994.

[17] G. Ruhe and M. O. Saliu. Art and Science of Software Release Planning. *IEEE Software*, 22(6):47–53, 2005.

[18] T. L. Saaty. Decision making — the Analytic Hierarchy and Network Processes (AHP/ANP). *Journal of Systems Science and Systems Engineering*, 13, 2004.

[19] M. O. Saliu and G. Ruhe. Supporting Software Release Planning Decisions for Evolving Systems. In 29^{th} *Annual NASA Software Engineering Workshop*. IEEE Computer Society, 2005.

[20] D. Spinellis. Silver Bullets and Other Mysteries. *IEEE Software*, 24, 2007.

[21] C. Wohlin and A. Aurum. What is Important when Deciding to Include a Software Requirement in a Project or Release? In *Proc. International Symposium on Empirical Software Engineering*. IEEE Computer Society, 2005.

[22] R. K. Yin. *Case Study Research: Design and Methods*. Sage Publications Inc., 2003.