

Rigorous Support for Flexible Planning of Product Releases — A Stakeholder-Centric Approach and its Initial Evaluation

Ville Heikkilä¹, Anas Jadallah², Kristian Rautiainen³, Günther Ruhe⁴

Helsinki University of Technology^{1,3}, University of Calgary^{2,4}

ville.t.heikkila@tkk.fi¹, agjadall@ucalgary.ca², kristian.rautiainen@tkk.fi³, ruhe@ucalgary.ca⁴

Abstract

This paper addresses the problem of product release planning in iterative product development. We propose a method which combines decision, process, and tool support. The method, which is called SCERP, facilitates the active involvement of stakeholders in the different stages of the planning process. SCERP is flexible in the number of stakeholders involved, in the number of releases, in the number and definition of planning criteria, and in the selection of the best plan out of a set of optimized alternatives. A proof-of-concept of the method is given by a case study of release planning for a tool called Agilefant, which is developed with a process partially based on Scrum. The benefits of the method as demonstrated by the case study are: (i) better decisions by the product manager by relying on more objective information, (ii) more transparency of release decisions, and (iii) efficient tool support accompanying the whole process.

Keywords:

Release planning, Decision support, Iterative development, Scrum, Case study.

1. Introduction

Decisions on processes, resources and tools are the crystallization points to achieve quality of software-dependent products and services. A major part of the responsibility of a product manager is to balance a variety of needs from markets and stakeholders, and align them into an optimized allocation of sparse resources [10]. In Scrum [25], a single product manager (called product owner) is responsible of communicating with stakeholders to identify which are the most profitable features to include in the next few releases. Under the pressure of time, size and complexity, the decision is often done ad hoc, mainly relying on intuition and experience. However, solely relying on intuition reduces the transparency and trustworthiness of decisions. Simultaneously, it

increases the risk of ignoring important factors resulting in less qualified decisions. We argue that the question is not whether to use either intuition or rigor and science. Instead, the challenge is to find a good synergy between the two approaches.

When a product has many stakeholders, face-to-face communication quickly becomes unfeasible and needs to be replaced with a more efficient way to gather, aggregate, and align stakeholders' needs [17]. Decision support systems (DSS) have been utilized in a wide variety of fields to assist the decision makers to make better decisions when the problem becomes wicked [19], semi or unstructured, or is in general hard to handle solely by humans.

The research presented in this paper applies the concepts of software engineering decision support to the context of planning product releases for iterative and incremental development when multiple stakeholders are involved. We propose a process with integrated tool and decision support, which emphasizes the role and active participation of multiple stakeholders. Specifically, we show a proof-of-concept of the release planning process by conducting a case study on its deployment for planning releases of Agilefant, a backlog management tool developed with a process partially based on Scrum.

Section 2 of the paper describes related work. Section 3 presents a short description of the formal approach to release planning taken in this paper. The stakeholder-centric release planning method called SCERP and its integrated tool support are presented in Section 4. The case study is described in Section 5, followed by the SCERP output and key findings presented in Section 6. Applicability and limitations are discussed in Section 7. A summary and outlook for future research is provided in Section 8.

2. Related work

Jung et al. [12] suggested a cost-value requirements analysis using mathematical programming. They applied a rigorous algorithm for solving the knapsack

problem [15] to decide which requirements should be taken into the next release. However, there is no stakeholder involvement in the prioritization or decision process.

Bagnall et al. [3] studied the problem of release planning with the scope of just the next release. They considered a model with a list of features that may depend on each other and a list of customers each with a weight and wish list. According to this method, features can only be offered if all the enabling features are provided. There is only limited involvement of the stakeholders in the prioritization of requirements. The goal is to maximize the weight of the total number of stakeholders being completely satisfied.

Van den Akker et al. [28] applied mathematical programming to provide a solution for the next release problem, where the main planning criterion is the projected revenue of the features. The stakeholders do not have a role in prioritizing the features. Besides that, it was left unclear how the stakeholders are involved in the process of determining the projected revenue.

Greer et al. proposed the EVOLVE [11] iterative approach for solving the release planning problem. The method requires the prioritization of features by all the selected stakeholders in terms of urgency and value. Besides that, it tries to balance the conflicting stakeholders' opinions to achieve the highest degree of satisfaction with the resources available. Although providing stakeholders the opportunity to prioritize features, they are no longer involved in the final process of selecting the release plan.

Karlsson et al. studied the problem of requirements prioritization as a pre-request for successful product release planning [14]. The authors conducted two experiments to investigate the differences between requirements prioritization techniques with and without tool support. Although the stakeholders are involved in the prioritization process, the research focuses on understanding differences in prioritization time-consumption, ease of use, and accuracy.

Wiegiers [29] proposes a process for prioritizing requirements based on value, cost and risk. The value of a feature is estimated by customer representatives based on relative benefit and relative penalty on a nine point scale. Relative cost and risk are estimated by developers using the nine point scale. A simple formula is then used to calculate the final priority value. Again, customers do not directly participate in the process.

Cohn [7] describes four techniques for requirements prioritization and a release planning method. The first technique is based on value, cost, learning, and risk of features. The second technique is based on financial measures calculated for the proposed features. Both of these techniques rely on the

product manager and internal stakeholders for input and customers are not directly involved. The third technique is based on the Kano-model [13] and suggests a customer survey to categorize features. However, feature priorities inside the Kano-categories are not addressed. The fourth technique is a version of Wiegiers' method described above. The proposed release planning method is a simple greedy algorithm.

In the Scrum software development process, the prioritization of features or user stories in the so called product backlog is the responsibility of the product owner [25]. Releases are then planned using a simple greedy algorithm by selecting the highest priority product backlog items into proposed releases [25]. One method for backlog prioritization is given by Schwaber [26]. He suggests that the product owner uses a ping-pong-ball method. The method is a variation of the hundred dollar test [16], which in turn is a variation of cumulative voting. Another example of prioritization and release planning in the agile software development is the planning game proposed by Beck [4].

Scrum and XP share at least two critical preconditions that are required for the planning methods to have a chance to work. First, there must be only one customer representative, and second, the customer representative must have sufficient knowledge of the requirements' priorities and time to participate in the planning. However, there are often many customers who have different or even contradictory needs, and establishing direct access to a customer representative may be challenging [5].

3. Problem statement

In this section, a formulation of the release planning problem as studied in this paper is given. A planning method based on this formulation is then provided in Section 4.

3.1. Features and related decision variables

This paper uses the concept of a "feature" as the basic unit for release planning. Features are the "selling units" or "minimum marketable features" [9] provided to the customer. In the context of this research, we follow the definition given by Wiegiers [29], which defines "a product feature as a set of logically related requirements that provide a capability to the user and enable the satisfaction of business objectives".

We assume a set of features $F = \{f(1), f(2), \dots, f(N)\}$. The goal is to assign the features to a finite number K of release options or to decide to postpone the feature. A release plan is characterized by a vector of decision variables $x = (x(1), x(2), \dots, x(N))$ with

$x(n) = k$, if feature $f(n)$ is assigned to release option $k \in \{1, 2, \dots, K\}$, and (1)

$x(n) = K+1$, if the feature $f(n)$ is postponed (e.g., not contained in one of the next K releases). (2)

3.2. Stakeholders

Stakeholders are very important for performing realistic release planning. An operational method for selection of stakeholders is described in [27]. We assume a given set of stakeholders $S = \{S(1), \dots, S(q)\}$. Each stakeholder $S(p)$ can be assigned a relative weight $\lambda(p)$ on an ordinal nine point scale ranging from extremely high (9) to extremely low (1). The weight is assigned based on the relative importance of the stakeholder.

3.3. Prioritization of features by stakeholders

In order to select and schedule features, there must be an agreed upon statement of priorities for features. In our model, prioritization by each stakeholder $S(p)$ can be done with respect to different criteria. We define them on an ordinal nine-point scale. Possible criteria for prioritization are overall business value, urgency (time dependency), dissatisfaction if feature is not included in a release, risk (using an inverted scale), frequency of use, etc.

3.4. Resource constraints

Different resources are required for the implementation of features, and there are capacity bounds on the amount of resources available in each release cycle. We consider R types of resources involved in the implementation of features. Correspondingly, we define resource capacities $Cap(r, k)$ for each resource type $r = 1, \dots, R$ and all releases $k = 1, \dots, K$. To become a feasible plan, decision variables must satisfy

$$\sum_{x(n)=k} resource(n, r) \leq Cap(k, r) \quad (3)$$

for all releases $k = 1, \dots, K$ and all resource types $r = 1, \dots, R$.

3.5. Objective function

The objective is the maximization of a function $F(x)$ among all release plans x satisfying the above technological and resource constraints (1) – (3). $F(x)$ is

composed of the weighted average priority vector $WAP(n)$ defined for each feature $f(n)$. Therein, the weighted average priority is a function including the different possible criteria. For each release option k , parameter $\xi(k)$ describes the relative importance of the release option and its relative impact to the objective function. For further details, see [19].

$$F(x) = \sum_{k=1 \dots K} \xi(k) [\sum_{n: x(n)=k} WAP(n)] \quad (4)$$

4. Stakeholder-centric release planning method SCERP

Understanding stakeholders, especially understanding user needs, is still one of the key success factors for software development [27][6]. The proposed stakeholder-centric release planning method provides rigorous process, tool, and decision support especially for situations where multiple stakeholders are involved. The key-steps of the method using the proprietary release planning tool ReleasePlanner™ [22] are described in the following sub-sections.

4.1. Step 1: Selection of critical stakeholders and pre-selection of candidate features

To increase the efficiency of the prioritization process, the method starts with a pre-selection step. This step aims at providing a deeper understanding of user requirements and to reduce the set of candidate features for prioritization in Step 2, which is along the lines of good requirements engineering practices [29]. The key parameters of the process step are given in Table 1.

Table 1. Key parameters for step 1

Purpose	Select critical stakeholders to discuss a reasonable set of validated candidate features for prioritization by all stakeholders.
Roles	Product manager and most critical stakeholders
Description: The idea is to pre-select relevant features to be studied later in more detail. For that, most critical stakeholders are invited to provide their inputs. This step simultaneously serves as a validity check for the understandability of the features under consideration.	
Input	Feature repository, set of all stakeholders
Output	Pre-selected candidate features

4.2. Step 2: Prioritization of features

The next step is to prioritize candidate features based on a set of agreed upon criteria. The tool offers

the possibility to assign stakeholders to groups of features and criteria. The prioritization itself is done using a nine-point scale ranging from extremely high (9) to extremely low (1). Prioritization can be either free or cumulative. Free prioritization allows stakeholders to give any priority to any feature, while cumulative prioritization forces stakeholders to select priorities between features. The key parameters of the process step are given in Table 2.

Table 2. Key parameters for step 2

Purpose	Get feedback from the stakeholders regarding their preferences.
Roles	Product manager and all nominated stakeholders
Description:	Each stakeholder S(p) is asked to prioritize for the assigned candidate features and evaluation criteria.
Input	Candidate features subset
Output	Priority profile and levels of conformance between stakeholders per feature

4.3. Step 3: Collective effort estimation

Similar to prioritization, stakeholders are nominated according to their role for the effort estimation process following some form of Delphi technique [23] or applying Planning Poker [20]. The key parameters of the process step are summarized in Table 3.

Table 3. Key parameters for step 3

Purpose	Obtain reasonable effort estimates for each feature.
Roles	Special set of stakeholders (typically developers)
Description:	Effort estimation per feature by nominated stakeholders
Input	Candidate features subset
Output	Effort estimates per feature.

4.4. Step 4: Calculation of optimized release plan alternatives

Instead of using the heuristic greedy algorithm, we apply the optimization-based planning method. Besides the guaranteed level of optimality, the advantage of this approach is that not just one, but a portfolio of optimized plans is offered. More specifically, the underlying algorithms are designed to generate five optimized and diversified release plan alternatives. For details on this we refer to [19]. The key parameters of Step 4 are summarized in Table 4.

Table 4. Key parameters process step 4

Purpose	Generating optimized release plan alternatives.
Roles	Product manager
Description:	See [19] for a description of the optimization algorithms.
Input	Stakeholders and their weights Candidate features set Features effort estimation and capacities per release Stakeholder's priority profile.
Output	Optimized release plan alternatives

4.5. Step 5: Prioritization of alternative plans

Based on the stated objectives, the optimized plans are all close to optimal and of almost the same high formal optimality (typically above 95%). However, formal optimality is not the only quality criterion considered. There is the assumption that stakeholders have different additional aspects to be looked at. For example, how cohesive the selected features of a release are considered or how risky their combined implementation is expected to be. These aspects are very hard to be modeled upfront. In the proposed SCERP method, selected stakeholders have the opportunity to prioritize between the proposed planning alternatives. The nine-point prioritization process is similar to prioritization among features as described in Step 2. The key parameters of the process step are summarized in Table 5.

Table 5. The key parameters of process step 5

Purpose	Obtain stakeholders feedback on generated alternatives.
Roles	Stakeholders and the product manager
Description:	(Selected) stakeholders are asked for their priorities for the five optimal alternative release plans.
Input	Optimized release plan alternatives
Output	Plan alternatives evaluation report and suggestion for final plan.

4.6. Tool support

One of the proposed key benefits of the SCERP method is its integrated tool support. For this, SCERP utilizes the ReleasePlannerTM [22] web-based decision support tool. The features of the tool are well aligned with the method:

- Since the tool is web-based, it allows stakeholders from different places and different time zones to be involved in the process.

- The tool allows the stakeholders to prioritize features based on a flexible number of criteria. This can be used for Steps 1 and 2.
- The tool facilitates flexibility as it allows context-specific definition of stakeholders, releases, features, and planning criteria, e.g. feature dependencies.
- The tool applies optimization algorithms to generate five diversified and optimized release plan alternatives (Step 4).
- The tool allows selected stakeholders to prioritize the alternative plan solutions (Step 5).

5. Case study background and design

An illustrative case study [30] was conducted to show how the SCERP method can be applied in a real software development project. In what follows, we provide some background information of the case study and details of the case study design. In Section 6, the SCERP method output and key findings are presented.

5.1. Background information

The SCERP process was applied to release planning of *Agilefant* [1], an open-source backlog management tool developed by the ATMAN research project of Helsinki University of Technology. Developed since 2006, Agilefant is currently used in several companies internationally and in Finland [1]. The development of Agilefant follows the Cycles of Control (CoC) framework [21], which is a framework for time-paced, iterative and incremental software development partially based on Scrum. The product manager in the case study is the product owner of Agilefant. All features that have been planned to be implemented in Agilefant are kept in a product backlog, which is a simple prioritized list of features.

5.2. Case study design

For the description of the case study design, we follow the process steps of SCERP. In addition to using the SCERP method, a manual release plan was created by the product owner for the purpose of comparative analysis. The manual plan was created according to Scrum release planning instructions [25].

5.2.1. Step 1: Pre-Selection of candidate features.

Pre-selection of candidate features was performed by the product owner of Agilefant. He conducted the selection based on his previous observations of the use of Agilefant in the two most important companies using Agilefant. Based on these observations, ten out of 73 backlog items were selected for the next steps.

5.2.2. Step 2: Stakeholder invitation and voting.

Stakeholders were invited by a news post in the homepage of Agilefant and from a list of known adopters. 33 stakeholders expressed their interest in the prioritization, of which 19 participated in the voting. Cumulative voting was applied in this case study as it is more likely to create more diversified results [29][8].

Three different prioritization criteria were selected:

1. Value (“What is the perceived (relative) value of the feature for you?”).
2. Urgency (“How urgently do you need the feature?”)
3. Dissatisfaction: (“How dissatisfied you would be if the feature becomes postponed”)

As the set of features and criteria was considered quite small, no special assignments were made and each stakeholder was asked to vote for the priority of every feature based on each criterion.

5.2.3. Step 3: Collective effort estimation. Using a formal method in this simple case was considered unnecessarily complicated, as there were only two active developers at the time. The resource consumption of features was estimated by the developers in an informal meeting. The product owner of Agilefant was also present to answer any questions about the features. One resource type (Full Time Equivalent developer (FTE)) was used.

5.2.4. Step 4: Calculation of the optimized release plan alternatives. Before the optimization algorithm could be used, the number of releases and releases’ resource constraints needed to be defined. The two next releases were selected as the scope of the planning. The first release was given relative importance value of nine and the second release was given five. The total effort capacity of both releases was determined to be 22 FTE-days. The tool was then used to generate five optimized release plan alternatives.

Table 6. Features, effort estimates and overall priority

ID	Effort	Feature	Overall priority (9-1) (n = 19)		
			Mean	STDV	Rank
F01	12	Limiting restricted user access to defined projects	4,66	3,06	6
F02	10	Future projects' planned sizes affect load calculations	4,78	2,87	5
F03	9	Daily Work for teams	6,44	2,49	1
F04	7	Exporting timesheet reports as CSV	3,25	2,45	10
F05	5	Exporting backlogs as CSV	3,62	2,39	8
F06	4	Story point estimation	5,97	3,07	2
F07	3	Transforming TODOs into new backlog items	5,44	3,00	4
F08	3	Logging spent effort for iterations	3,57	2,22	9
F09	2	Enhancements for displaying themes in the product backlog view	4,12	2,86	7
F10	2	Hiding past projects and iterations	5,92	2,81	3

5.2.5. Step 5: Prioritizing the release plan alternatives. Prioritization of the release plan alternatives was done by evaluating how well each release plan matched the needs of ten stakeholders based on a nine-point scale. Stakeholders were not shown the degree of formal optimality of the plans. Five of the nine values were given textual descriptions, which were “Perfect match”(9), “Good match”(7), “Okay match”(5), “Slight match”(3) and “No match”(1). In addition to the five optimized plans, the manual plan created by the product owner was included for prioritization.

5.3. Survey

To provide further insight into the results, the participants were also asked to fill a survey. The survey gathered additional information about the participants and their attitudes towards the planning method. The stakeholders were instructed to answer the survey after they had performed the voting in ReleasePlanner™. The survey was implemented with a web-based survey software. The survey contained free text fields and six-choice attitude scale questions. The extreme values in the attitude scale were named “Strongly agree” and “Strongly disagree”. The choices in between the extreme values on the scale were not named. A “Don’t know” choice was also provided. In the analysis of results, “Strongly agree” is given score 6 and “Strongly disagree” score 1. The values in between are interpreted as follows: “Agree” (5), “Slightly agree” (4), “Slightly disagree” (3) and “Disagree” (2).

The survey contained one attitude scale statement per feature in the form “I understood the feature X” (Q1-Q10). Other included attitude scale statements were “This method of prioritization allows me to express my needs” (Q11), “I understood what I was expected to do” (Q12) and “The list of features

contained the features that are most important for me” (Q13). A free text field for recording how much time the voting took was also included (Q14). An additional free text field was provided for general feedback about the prioritization method (Q15).

6. SCERP output and key findings

The output results produced by the SCERP method in the case study are presented in Section 6.1. Key findings of the case study are described in Section 6.2.

6.1. SCERP output

Table 6 summarizes the results of the feature prioritization. The “Feature” column lists the ten features pre-selected for prioritization in case study Step 1. Ultimately, 19 stakeholders voted for the priority of these features in ReleasePlanner™. The “Effort” column shows the effort estimates of the features, which were created in Step 3. Results of the prioritization voting in Step 2 are shown in the columns under the “Overall priority” heading. The “Mean” column shows the mean of the vote values of each feature calculated over all stakeholders and criteria. Standard deviation of the votes and rank order based on the means are also shown.

Table 7 shows the release plans created in case study Step 4. The columns O1-O5 contain the optimized plans and column M contains the manual plan. Each F-row shows the assignment of a feature to a release in the different plans. One signifies that the feature is scheduled to the first release, two signifies the second release, and three signifies that the feature is postponed. The “Opt (%)” row shows the degree of formal optimality of each plan.

Table 8 shows the results of the release plan prioritization in case study Step 5. Each row shows the match scores given by one of the ten stakeholders who participated in this step. The sum of values, median

and standard deviation of the match scores of each plan are also provided.

Table 7. Release plan alternatives

ID	O1	O2	O3	O4	O5	M
F01	2	2	3	2	2	3
F02	3	3	2	2	3	1
F03	2	2	1	3	1	2
F04	3	1	2	3	3	2
F05	1	3	2	1	1	3
F06	1	1	1	1	1	1
F07	1	1	1	1	2	1
F08	1	1	3	1	2	2
F09	1	1	1	1	1	1
F10	1	1	1	1	1	1
Opt. (%)	100,0	98,9	98,6	97,1	96,8	93,4

Table 8. Release plan alternative voting

Stakeholder	Release plan					
	O1	O2	O3	O4	O5	M
S01	4	3	9	7	1	2
S02	7	5	5	7	7	4
S03	2	2	8	1	7	3
S04	6	9	9	5	8	8
S05	3	6	3	3	5	3
S06	6	5	4	7	5	2
S07	4	4	5	7	4	4
S08	3	2	5	1	6	4
S09	5	4	7	3	6	8
S10	2	3	9	1	5	8
Sum	42	43	64	42	54	46
Median	4,0	4,0	6,0	4,0	5,5	4,0
STDV	1,8	2,1	2,3	2,7	2,0	2,5

6.2. Key findings

This section describes the key findings. Survey results are also introduced in this section.

6.2.1. Stakeholders' priorities varied considerably.

There were substantial differences between stakeholders' opinions concerning the priorities of the features. This was measured by the standard deviation of the stakeholders' votes. The results are shown in Table 6. We argue that it is important to look at not just one, but a portfolio of stakeholders.

6.2.2. Stakeholders understood the features. The results of the planning can only be as good as the stakeholders' understanding of the features. In the results of the survey questions Q01-10, five features

got median of six (excellent understanding), four features got median of five (very good understanding) and one feature got median of four point five (between good and very good understanding). The scale was from one to six. The standard deviations of the answers varied between 0,50 and 1,42. The results show that the stakeholders thought that they understood most of the features very well based on the descriptions provided in the tool.

6.2.3. Stakeholders understood the prioritization method. Planning success depends on whether the stakeholders understand what they are expected to do. The Median answer to survey question 12 (Q12) was 5 ("Agree") with the standard deviation being 1,00. This implies good understanding of the prioritization method.

6.2.4. Stakeholders liked the prioritization method. We also investigated the acceptance of the web-based prioritization process. The stakeholders thought that the method of prioritization was quite good (Q11). The Median answer was 4.5 (between "Agree" and "Slightly agree") and the standard deviation 1.24.

6.2.5. The SCERP method is time-efficient for stakeholders. On the average, the stakeholders spent approximately 14 minutes performing the prioritization in ReleasePlanner™ (Q14). The longest time spent was 30 minutes and the shortest time was 5 minutes. With three criteria and ten features, the average time taken to vote for one feature on one criterion was approximately 30 seconds.

However, the method required considerable effort from the product owner. Estimated effort taken to complete the release planning was approximately two complete workdays. This includes stakeholder invitation, pre-selection of candidate features, writing feature descriptions, criteria selection, and all other related tasks.

6.2.6. The list of candidate features was insufficient.

The results from survey question 13 (Q13) reveal that stakeholders thought that the list of candidate features did not contain the features that were most important to them. The median answer of this question was 2 ("Disagree") and standard deviation 1,32. This result is supported by the free feedback from the survey (Q15). Two stakeholders commented that they would have liked to suggest other features to be added to the list of candidate features.

6.2.7. Optimized plans are more optimal than the manual plan. Using the manual plan generated by the product owner as the baseline, we have measured the

quality of the suggested plans. For that purpose, the quality was measured by the overall utility function used for the optimization [19]. It is a linear function aimed to maximize overall stakeholder satisfaction. When we compare the optimized plans O1 to O5 with the manual plan (M) we observe that all optimized plan alternatives are formally better than the manual one. The plans and their degree of optimality are summarized in Table 7.

6.2.8. Selection and definition of prioritization criteria must be done carefully. The differences of the vote means between the three criteria were very small in this case study. This would suggest that the criteria were not chosen optimally or defined clearly enough. This assumption is further supported by the results of the survey (Q15). Three stakeholders commented that they had difficulties differentiating between the three criteria.

6.2.9. Optimized plan alternative is more acceptable than the manual plan. Using the manual plan generated by the product owner as the baseline, we can observe that two of the generated plans have better acceptance among the stakeholders. Increasing acceptance by offering a portfolio of qualified alternates is the key idea behind Step 5 of SCERP. Incorporating all stakeholders' inputs results in a preference for O3, as seen in Table 8.

7. Discussion

The initial results and the feedback gained from the survey are very encouraging. The results of the case study suggest that SCERP works well. There are several factors that support this. First, the stakeholders expressed in the survey that they liked the method, even if they thought that the most important features for them were not included in the voting. One possible explanation for this is the transparency of the release planning process that is provided by SCERP and its tool support, and the possibility to contribute to the planning. This is a confirmation of similar former industry experience [2][18].

A second explanation for the acceptance is the fact that SCERP did not require too much spent effort from the stakeholders, while still providing a rigorous process. This adds to the flexibility of performing the prioritization anywhere and anytime.

Third, while the standard deviation for the votes on each separate feature was very high, the voting between the release plan alternatives revealed a clear favorite. Without this step (Step 5), which is a novel contribution of SCERP, the choice between the seemingly good release plan alternatives would have

been purely based on intuition. Also, the high standard deviation of the votes means that an optimal plan is hard to find by intuition alone. This merits the use of decision support tools to allow taking into account the diversified needs of the stakeholders.

Fourth, the prioritization process created clear support for one of the generated release plan alternatives, which encouraged the product owner of Agilefant to choose the plan instead of his own manual plan. While the manual plan was based on the product owner's interpretation of the acute needs of the two most important companies using Agilefant, voting for the release plan alternatives proved that there was another option that was a better compromise when taking all stakeholders into account.

Without the rigorous process and the transparency and wealth of information provided by the tool support, the product owner would have chosen his own, slightly sup-optimal plan alternative having lower overall acceptance. If such sup-optimal choices would be made for each release, we can argue that the acceptance of the product would suffer a lot during its lifetime.

In our case study we applied SCERP for release planning in a small project which uses an agile development process. However, this does not limit the use of SCERP into this context. In fact, in this very simple case we did not use nearly all features of ReleasePlanner™ that were available. That notwithstanding, we still got good results. In a more complex case the diversity and flexibility of ReleasePlanner™ would allow us to fine tune the prioritization process. Also, in a more complex situation, the limitations of relying only on human intuition would be far greater (see also [2][18]).

While participating in the voting did not take much effort, preparing the features for voting did. The reason is that the features in Agilefant's product backlog were not originally written with SCERP in mind and it took several hours to prepare for the voting. Ideally, product backlog management procedures should be compatible with the prioritization process, and there should be an easy way to export/import backlog items between the different tools involved.

The stakeholders in this case study did not find the list of candidate features sufficient. This result might be caused by several factors. First, the list of candidate features might have been too short. Second, the pre-selection of candidate features might not have been performed with the most representative stakeholders. The pre-selection step of SCERP should probably have included a wider stakeholder base and even some simple pre-voting might have been used. Third, the initial list of 73 features might not have contained the most important features to begin with. Using tool

support does not remove the need to use proper requirements engineering techniques, including face-to-face time with customers to elicit their real needs.

There are several threats to validity. As in any case study, there is no claim for external validity. Substantial further research is needed to demonstrate the scalability of SCERP. Another threat comes from lack of complete adherence to the defined SCERP process. The case study was running concurrently with the development of SCERP. Especially Step 1 should have been performed under broader stakeholder involvement. Also, the SCERP process depends on powerful release plan optimization algorithms, which in this case ties it to a specific tool. All other steps can be performed without the specific tool support, and in theory, other release plan optimization algorithms could be used.

Finally, selection of prioritization criteria is a crucial issue. It was observed that the priorities assigned to features were rather similar for all three criteria. This raises the question if the criteria were well enough understood and/or followed. More systematic preparation would have ensured that the criteria were properly defined and aligned with business objectives, i.e. stakeholders could have been asked for early validation of the criteria.

8. Summary and future research

We have presented a stakeholder-centric release planning method called SCERP for supporting product release decisions. We consider four main contributions in the paper:

- SCERP combines the strengths of rigorous processes with the key principle of keeping processes flexible.
- SCERP is designed as decision, process and tool support for release decisions to be made by the product manager.
- The method allows active participation of all nominated stakeholders in the planning process, including the decisions about the final release plan.
- Promising initial evaluation of the method by a real software development project.

From offering systematic process and decisions support based on an existing tool, we have promising initial evidence that the product manager can better perform his coordinating and decision role.

Despite all the encouraging initial results, we do not claim external validity of the results, but just the “Proof-of-concept” of the method. While the whole method is designed to be scalable, a proper validation of this capability is outstanding.

Besides the need for more comprehensive validation towards the method’s scalability, other topics of future research do exist. The SCERP process steps are planned to be closer linked to Scrum process steps to make the approach a direct Scrum release planning tool support. This would also facilitate scaling Scrum to enterprise level and support coordinated and comprehensive involvement of stakeholders in the whole release planning process. Direct data transfer from a Scrum backlog management tool to ReleasePlanner™ would enable frequent re-planning with up-to-date estimates and remove some of the set-up overhead of SCERP.

9. Acknowledgments

Thanks to Tekes (the Finnish Funding Agency for Technology and Innovation) and all the Finnish companies participating in the ATMAN research project for financing the research. We are especially grateful to all the stakeholders who actively participated in the case study.

One of the authors would like to thank the Canadian NSERC Natural Sciences and Engineering Research Council (Discovery Grant 250343-07) for the financial support of this research.

10. References

- [1] Agilefant, <http://www.agilefant.org>
- [2] Amandeep, A., Ruhe, G. and Stanford, M., “Intelligent Support for Software Release Planning”, in Proc. 5th International Conference on Product Focused Software Process Improvement, Kansai, Japan, 2004, pp. 248-262.
- [3] Bagnall, A.J., Rayward-Smith, V.J. and Whittle, I.M., “The Next Release Problem”, Information and Software Technology, vol. 43, 2001, pp. 883-890.
- [4] Beck, K., “Extreme programming explained : embrace change”, Addison-Wesley, Reading, MA, 2000.
- [5] Cao, L. and Ramesh, B., “Agile requirements engineering practices: An empirical study”, IEEE Software, vol. 25, 2008, pp. 60-67.
- [6] Chow, T. and Cao, D., “A survey of critical success factors in agile software projects”, Journal of Systems and Software, vol. 81, 2008, pp. 961-971.
- [7] Cohn, M., “Agile Estimating and Planning”, Prentice Hall, Upper Saddle River, NJ, 2006.
- [8] Davis, A. M., “Just enough requirements management: where software development meets marketing”, Dorset House Publ., New York, NY, 2005.

- [9] Denne, M., Cleland-Huang, J. "The incremental funding method: Data-driven software development", IEE Software, vol. 21, 2004, pp. 39-47.
- [10] Ebert, C., "The Impacts of Software Product Management", Journal of Systems and Software, vol. 80, 2007, pp. 850-861.
- [11] Greer, D. and Ruhe, G., "Software Release Planning: An Evolutionary and Iterative Approach", Information and Software Technology, vol. 46, 2004, pp. 243-253.
- [12] Jung, H. W., "Optimizing value and cost in requirements analysis", IEEE Software, vol. 15, 1998, pp. 74-78.
- [13] Kano, N., Seraku, N., Takahashi, F., Tsuji, S., "Attractive Quality and Must-Be Quality", translated from Japanese, Hinshitsu (Quality, The Journal of the Japanese Society for Quality Control), vol. 14, 1984, pp. 39-48.
- [14] Karlsson, L., Thelin, T., Regnell, B., Berander, P., Wohlin, C., "Pair-wise comparisons versus planning game partitioning-experiments on requirements prioritisation techniques", Empirical Software Engineering, vol. 12, 2007, pp. 3-33.
- [15] Kellerer, H., Pferschy, U., Pisinger, D., "Knapsack Problems", Springer, 2003.
- [16] Leffingwell, D., Widrig, D., "Managing software requirements: a unified approach", Addison-Wesley, Reading, MA, 2000.
- [17] Moe, N. B. and Aurum, A., "Understanding Decision-Making in Agile Software Development: a Case-study", Proc. 34th Euromicro/SEAA, 2008, pp. 216-223.
- [18] Momoh, J. and Ruhe, G., "Release planning process improvement - An industrial case study", Software Process Improvement and Practice, vol. 11, 2006, pp. 295-307.
- [19] Ngo-The, A., Ruhe, G., "A Systematic Approach for Solving the Wicked Problem of Software Release Planning", Soft Computing, vol. 12, 2008, pp. 95-108.
- [20] Ostvold, K., Haugen, N., Benestad, H., "Using planning poker for combining expert estimates in software projects", Journal of Systems and Software, vol. 81, 2008, pp. 2106-2117.
- [21] Rautiainen, K., Vuornos, L., and Lassenius, C., "An experience in combining flexibility and control in a small company's software product development process", Proc. Symposium on Empirical Software Engineering, 2003, pp. 28-37.
- [22] ReleasePlanner™, <http://www.releaseplanner.com>
- [23] Rowe, G., Wright, G., "The Delphi technique as a forecasting tool: issues and analysis", International Journal for Forecasting, vol.15, 1999, pp. 353-375.
- [24] Ruhe, G., "The Art and Science of Product Release Planning", forthcoming, CRC Press, 2009.
- [25] Schwaber, K., Beedle, M., "Agile Software Development with Scrum", Prentice Hall, Upper Saddle River, NJ, 2001.
- [26] Schwaber, K., "The enterprise and Scrum", Microsoft Press, Redmond, WA, 2007.
- [27] Sharp, H., Finkelstein, A. and Galal, G., "Stakeholder Identification in the Requirements Engineering Process", Proc. 10th International Workshop on Database and Expert Systems Applications, Florence, Italy, pp 387-391.
- [28] Van den Akker, M., Brinkkemper, S., Diepen, G. and Versendaal, J., "Software product release planning through optimization and what-if analysis", Information and Software Technology, vol. 50, 2008, pp. 101-111.
- [29] Wiegers, K., "Software Requirements", Microsoft Press, Redmond, WA, 2003.
- [30] Yin, R. K., "Case Study Research: Design and Methods", Thousand Oaks, CA, Sage Publications, 2002.