Releasing Sooner or Later: An Optimization Approach and Its Case Study Evaluation

Jason Ho
Department of Computer Science
University of Calgary
Calgary, Alberta, Canada
hott@ucalgary.ca

Guenther Ruhe[1)2)]
[1)]Department of Computer Science
University of Calgary
[2)]Expert Decisions Inc.
Calgary, Alberta, Canada
ruhe@ucalgary.ca

***Abstract*—Decisions about the release date need to balance between the degree of readiness (quality) of the product and the potential competitive advantage and added value of (early) delivery. Based on an existing optimization approach for solving the maximum value release planning problem for a fixed release time, we provide a re-optimization approach for which includes local and global re-planning exchange operations to determine the outcome of varying the release dates. The proposed method determines a set of trade-off solutions related to (i) the value of the features implemented, (ii) their estimated quality, and (iii) the release time. The approach is evaluated by a real-world case study taken from an ongoing project which is to develop the new main release 2.0 for the release planning decision support platform ReleasePlanner™ v1.6.**

***Index Terms*—Release engineering, when-to-release, decision support, software quality, case study.**

I. Introduction

Deciding about the proper release time is of key importance for the success of a product. The product manager has to balance between offering the release product early and providing a product of highest quality (which typically takes more time). In this paper, we study when-to-release decisions and propose an approach to create a set of trade-off solutions.

The focus of this (ongoing) research is on two answering to questions (RQ's):

RQ1: Given a specific release date, by varying release duration, how can we identify optimized features sets being trade-off solutions between value and quality?

RQ2: How can the process of creating trade-off solutions be tool supported?

For the first question, we propose an approach being an enhancement of an existing method called RASORP [9]. For the second question we perform a case study. In Section II of the paper, we briefly discuss related work. Problem definition and the proposed solution approach are the content of Section III. The case study is presented in Section IV, followed by some discuss and outlook on future research.

II. Related Work

Requirements engineering is a decision-centric discipline. In general, when-to-release decisions are often made ad-hoc based on business needs and project manager's experience. If software is released too early, it may not meet the quality benchmark and market readiness. However, if testing takes too much time, the product may go over budget and miss the window of opportunity [7]. Therefore, balancing between the costs of delivery with the value of release has been of interest.

McElroy&Ruhe [5] studied when-to-release decisions. They allowed (i) time-dependent value functions for each feature, (ii) flexible release dates varying within a given interval, and (iii) adjusted time-dependent resource capacities and applied genetic algorithms to determine value-risk tradeoff solutions.

Reliability is often measured based on remaining defects in the system. As software systems get more complex, completely removing all defects is very difficult. In an effort to quantify the reliability of a software system, a number of software reliability models (SRMs) have been used for making release decisions. SRM's that describe software failure occurrence are called software reliability growth models (SRGM). During testing, the SRGM describes the relationship between the cumulative number of detected software faults and the time span of testing [10]. Existing SRGMs have underlying assumptions that are largely violated in practice, which makes these models often inappropriate for estimating remaining defects (or failures).

III. Problem Definitions & Proposed Solution

A. Software Release Planning

A product feature is a set of logically related requirements that provide a capability to the user and enable the satisfaction of business objectives [13]. Release planning as addressed in this paper is the problem of assigning the optimized set of features into a release and providing an operational plan for their implementation with the human resources available and in fulfillment of the given technological constraints. For this problem, a fixed release date is assumed to be given. The problem studied in this paper is to determine the impact of varying this release date.

B. Value of Features

A feature value is evaluated based on the capability to enable users and satisfy business objectives. For our purposes,

value is determined by a stakeholder satisfaction-based scoring system implemented in ReleasePlanner™ v1.6[1]. According to that, all involved stakeholders prioritize features against specified criteria (such as value, risk, frequency of use) based on a nine- point scale [11]. An overall feature value called *value(n)* is determined as the weighted (over all criteria and all stakeholders) average of feature priorities.

Definition 1: For a feature set F1 expressing the features to be implemented in the up-coming release, the *Total Release Value* TRV(F1) is defined as

$$TRV(F1) = \sum_{f(n) \text{ from } F1} value(n) \quad (1)$$

C. Quality of a Release

Quality of a feature is hard to evaluate. To simplify the process, we assume that quality to be directly proportional to the amount of effort invested in integration testing and user acceptance testing, and the effort of developers to fix any defects found from testing. For that purpose, we apply the Cost of Conformance and Cost of Lack-of-Conformance measure introduced by Campanella [4].

Definition 2: For a feature f(*n*), *Cost of Conformance CoC(n)* is defined as the cost for re-running integration and testing effort through user acceptance test. Conversely, *Cost of Lack of Conformance CLoC(n)* is the cost for fixing defects detected during testing.

Definition 3: The *Cost of Quality* (CoQ) of a feature f(*n*) is calculated by:

$$CoQ(n) = C_oC(n) + CLoC(n) \quad (2)$$

Definition 4: If the total effort CoQ(n) is invested, feature f(n) will have very high VH quality level (100%). Correspondingly, if quality is at least 95% and 85% of CoC(n), then we will talk of High H and Good G quality level, respectively.

Definition 5: For a feature set F1 expressing the features to be implemented in the up-coming release, the *Total Release Quality* is defined as

$$TRQ(F1) = \sum_{f(n) \text{ from } F1} CoQ(n) \quad (3)$$

D. Problem Statement

Definition 6: The *release planning problem RPP* consists of finding an operational plan implementing feature set F1 out of a given feature set F under

i. a set of technological constraints TC which have to be satisfied for implementation and testing;
ii. a pool of developers with individual capabilities to perform the implementation and testing tasks and
iii. a fixed release date RD for having done all the implementation and testing of features from F1.

Release engineering, among others, encompasses the question to find the most appropriate release date in consideration of quality and value. In order to make this decision, different scenarios for release time and quality need to be evaluated pro-actively. This is the problem called *W2RP*:

Definition 7: The *when-to-release problem W2RP* is defined by a sequence $\{RPP_i\}$ of problems RPP. Each individual RPP has a different fixed release date RD_i. W2RP means to determine operational release plans with varying feature sets F_i which are trade-off solutions[2] among all the variations of possible plans in terms of the three criteria

i. Maximize $TRV(F_i)$
ii. Maximize $TRQ(F_i)$
iii. Minimize RD_i

E. Solution Approach

The solution approach for W2RP is relying on two main components:

- A component (called RASORP and described in [9], [11]) able to solve the (operational) RPP
- A local re-planning procedure able to re-balance efforts between CoC(*n*) + CLoC(*n*) of features to be switched between varying release times.

The workflow of our proposed approach is given as Figure 1. Initially, a baseline release plan (from what is called *strategic release planning* or *road-mapping,* see [12]) is determined from using ReleasePlanner™ applying the baseline release time. *Local re-planning* is applied to accommodate for local changes of the release time. These local variations will only affects features that are closer to the release dates. This ensures consistency, and allows product managers to refer to the alternatives even after operation has started.

In order to also consider more substantial (global) variation of the release time, RASORP is applied for re-running RPP for alternative release time scenarios.

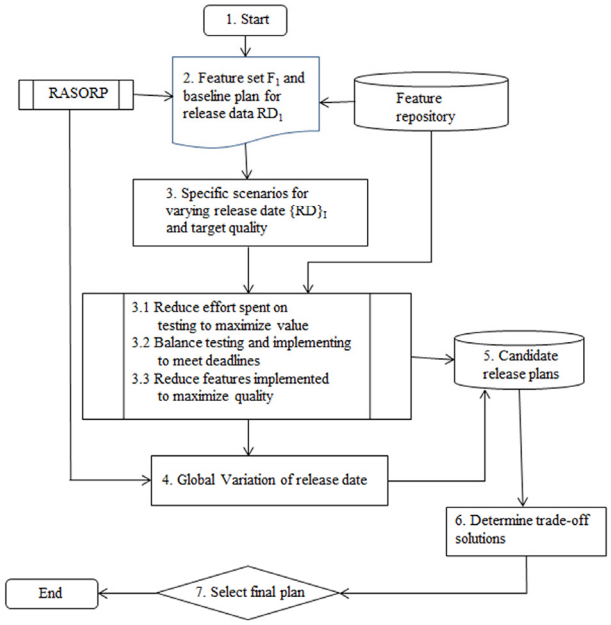

Fig. 1. When-to-release evaluation process

[1] www.releaseplanner.com

[2] Trade-off solutions refer to Pareto-optimal solutions for multi-criteria release planning, as defined in [11].

Overall, this process creates a pool of operational release plans. The ones being not dominated by any other plan are considered to be the decision support recommendation given to the product manager (being the Pareto-optimal solutions). The application of this process will be illustrated in Section IV by a real-world case study.

IV. Case Study

A. The Project

For illustration and initial evaluation purposes, a case study of a real life technology project is reported. The ongoing project is referring to the development of release version 2.0 for the decision support platform *ReleasePlanner*™ of *Expert Decisions Inc*. ReleasePlanner™ v1.6 is a proprietary product which has proven successful in numerous industrial and academic release planning projects (see [6] for the most recent example]).

The project considers 22 features that are included in the baseline release. These features were selected from a total set of 40 features which were collected in recent years from new feature requests, change requests, and innovative new ideas to be implemented. The topic discussed in this paper is an example for the latter category. All project's features, values and optimization results are available online [14].

There are seven *developers* working on the project. Each developer is assigned a portfolio of possible *tasks* (e.g., implementation, integration testing, acceptance testing, etc.). There are precedencies and dependencies between these tasks. For instant, testing can only start after implementation and can only finish after implementation is done. RASORP [9] has the capability to resolve complex dependencies and assign developers based on their productivity. Depending on the degree of usage of developers for the different types of tasks ranging from implementation to fixing defects, there will be a variation in the degree of quality of the feature.

Using RASORP, a baseline operational plan for how to implement $F_{1\ is}$ generated. The resulting release has duration $RD_1 = 77$ days with Total Release Value $TRV(F_1) = 137$. Based on a set of proposed release date variations, the authors study the impact of local and global variations around this plan.

B. Planning Scenarios

In the n order to generate trade-off solutions, we will explore the different distribution of effort for implementation of new features and fixing defects for feature-in-progress. For example, if a developer has to fix a critical defect, which costs 20% more time than previously planned (increase in CoQ), their capability for fixing defect is reduced to only 80%.

In Fig. 2, which is a simplified diagram of how local variations are done through re-allocating developers' time, we explain how Step 3 in the solution approach is executed. When time is shortened, either new features will be delayed (lower TRV), or defects remain unfixed (lower TRQ). However, stability is maintained by preserving the features' order in baseline plan.

In this example, the release date is shortened from 60 days to 57 days. Firstly, to maintain the stability to the plan, feature "Prioritization" remains unchanged. Only feature "Reporting", valued at 1, and feature "Analysis", valued at 9, will be re-allocated. Secondly, based on solution approach, we have the following three variations:

1. To maximize TRQ, both features "Reporting" and "Analysis" will be dropped, as there is not enough effort for testing and fixing all defects. In this variation, all features will have Very High (VH) quality (Q=100%). However, TRV is reduced as the plan has fewer features.
2. To minimize RD, features closer to completion will be prioritized. As "Reporting" is closer to release date RD (57 Days), testing effort for "Analysis" will be re-allocated to complete "Reporting". This results in a balance plan with High (H) quality (Q=95%) and moderate values. Some defects may be fixed later.
3. To maximize TRV, features with critical business values will be prioritized. As feature "Analysis" has higher value, testing effort for "Reporting" will be compromised to focus on finishing "Analysis". This plan will have the most values, while the quality may be compromised for low valued features.

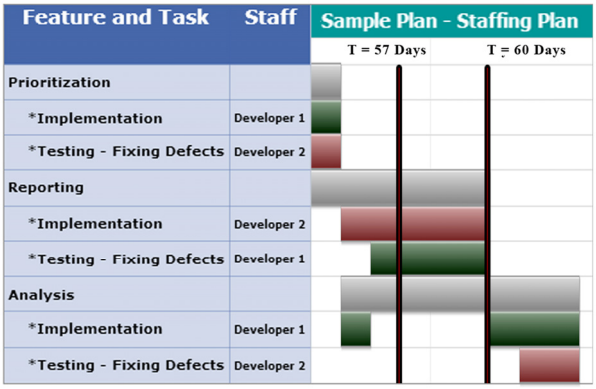

Fig. 2. Examples of local variation by scenarios.

The local variation of effort details with actual effort (in person days) is available online [14].

C. Analysis and Results

Upon running global variation and optimization using RASORP, many trade-off solutions of the release plan were selected based on *Definition 7*. Figure 3 represents the potential solution sets, with clusters of very high quality solutions (Q = 100%), high quality (Q ≥ 95%), and acceptable quality solution (Q ≥ 80%). All potential solution sets are (i) satisfactory of the planning requirements for the release and (ii) corresponding to the projected quality and values based on the local variations.

Three alternative trade-off solutions were finally offered to human experts for decision- making:

Table 1 Recommended solutions for human experts

	Quality	Values	RD	Features
Alternative 1	100.00%	107	50	17
Alternative 2	98.18%	124	63	21
Alternative 3	84.32%	137	70	22

Alternative 1 of VH quality and reached within the shortest release date (50 days). The only drawback is there are only 17 features in the release, with TRV of 107.

Alternative 2 is the balanced solution. It has H quality and high TRV. With 63 days duration, the RD is only 3 days longer than the targeted 60 person days in release duration.

Alternative 3 has the maximum TRV (137) and comprises of 22 features. However, it requires 10 additional days (compared to the targeted 60 person days) and accepting some compromise in quality (TRQ = 84.3%).

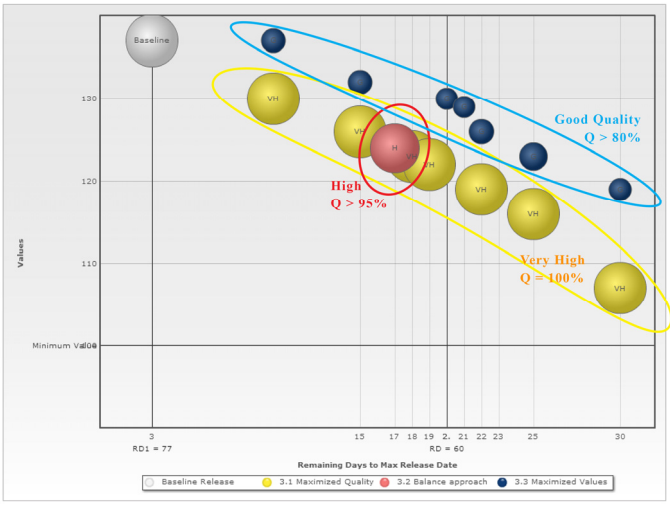

Fig. 3. Trade-off solutions for proposed scenarios.

D. Threats to Validity

Even though the results were applicable and useful for the case study performed, no claims can be made on external validity. One key threat to construct validity of this approach is the modeling and measurement of quality and value. Quality modeling and qualification is complex, especially in the case of integrated testing scenarios of large systems. The current quality prediction is a simplification of reality. The actual method to be selected will depend on the quality criteria selected (e.g., reliability versus security versus performance).

The usefulness of the results will depend on the scenarios elaborated. The method presented is flexible as the user is able to define relative target quality levels. The same is true for the possible variations of local and global release date changes.

V. Conclusions and Future Work

The results presented in this paper reflect the status of ongoing research and implementation. Performing when-to-release decisions in consideration of quality and testing criteria is considered a promising way for delivering decision support for product managers. The ideas outlined in the paper will be implemented as one part of the in the upcoming tool suite investigated in the case study of Section IV.

Comparisons with other methods could not be done, as there are no approaches known to us. However, further evaluation is needed to validate the usefulness of the proposed approach. Integrated tool support needs to be developed allowing pro-active evaluation of different release dates and its implication on predicted value and quality. Moving forward, the tool will incorporate test scenarios and defects data of both individual features and the integrated system. This is an extension from just effort and capacity estimation of the Quality model.

Re-optimization methods are supposed to increase efficiency of computations. Instead of performing the solution of a W2RP each time from scratch, algorithms will try to utilize the former solution as a starting point for solving the slightly revised problem.

The scalability of the method needs to be evaluated. With larger feature sets, the computational effort will grow exponentially. With the development and adoption of cloud and mobile technology, this proposed methodology`s calculations and optimizations can be done upfront and offline for being utilized online, on mobile platforms. This is relevant as the computations for resource allocation and when to release are complex as the number of features and staffs grow.

Acknowledgments

This research was supported by the Natural Sciences and Engineering Research Council of Canada, NSERC Discovery Grant 250343-12 and by the Microsoft Software Innovation Foundation SEIF award 2012. We also would like to thank Kornelia Streb for her technical support in this paper.

References

[1] A. Al-Emran, A. Jadallah, E. Paikari, D. Pfahl, G. Ruhe, “Application of Re-estimation in Re-planning of Software Product Releases”, in *Proc. ICSSP*, 2010, pp. 260-272.

[2] A. Aurum, C. Wohlin, “The Fundamental Nature of Requirement Engineering Activities as a Decision-Making Process,” *IST*, vol. 45, 2003, pp. 945-954.

[3] B. Boehm, V. R. Basili. "Defect Reduction Top 10 List”, *Computer,* vol. 34, no. 1, 2001, pp. 135-137.

[4] J. Campanella, "Principles of Quality Costs: Principles, Implementation and Use" ASQ Quality Press, 1999.

[5] J. McElroy, G. Ruhe, “When-to-release decisions for features with time-dependent value functions”, *Requirements Engineering Journal,* vol. 15 , 2010, pp. 337-358.

[6] G. Zorn-Pauli, B. Paech, T. Beck, H. Karey and G. Ruhe, “Analyzing An Industrial Strategic Release Planning Process – A Case Study at Roche Diagnostics,” Proc. REFSQ, 2013.

[7] R. Lai, G. Mohit, and P. K. Kapur. "A Study of When to Release a Software Product from the Perspective of Software Reliability Models." *Journal of Software,* vol. 6, 2011, pp. 651-661.

[8] S. McConnell, “Code Complete”. Microsoft Press, 2004.

[9] A. Ngo-The, G. Ruhe, "Optimized Resource Allocation for Software Release Planning", *IEEE TSE,* vol. 35, 2009, pp. 109-123.

[10] H. Ohtera, S. Yamada, "Optimum Software-Release Time Considering an Error-Detection Phenomenon During Operation," *IEEE Trans. on Reliability*, vol. 39, 1990, pp. 596-599.

[11] G. Ruhe, “Product Release Planning: Methods, Tools and Applications,” CRC Press, 2010.

[12] M. Svahnberg, G. Tony, R. Feldt, R. Torkar, S. B. Saleem, and M. U. Shafique, "A systematic review on strategic release planning models," *IST,* vol. 52, 2010, pp. 237-248.

[13] K. E. Wiegers, "Software requirements," Microsoft Press, 2009. https://sites.google.com/site/trongtanho/currentresearches, Research case study data, last accessed Feb 2013.