

SIMULATION-BASED UNCERTAINTY ANALYSIS FOR PLANNING PARAMETERS IN OPERATIONAL PRODUCT MANAGEMENT

A. Al-Emran, K. Khosrovian, D. Pfahl, G. Ruhe

**University of Calgary
Schulich School of Engineering
Department of Electrical and Computer Engineering
Calgary, Alberta T2N 1N4, CANADA**

ABSTRACT

Software product management takes place on strategic and operational levels. Strategic product management aims at assigning features to subsequent product releases such that technical, resource, risk and budget constraints are met. Operational product management focuses on the realization of a single software release. Its purpose is to assign resources to feature development tasks such that total release development time is minimized under given process and project constraints.

In this paper, we investigate the impact of uncertainty in estimates of developer productivity and feature-related effort on the structure and performance of operational release plans (ORPs). More precisely, we study the impact of variations of these estimates on (i) the duration of a single release (ii) the stability of the assignment of developers to tasks, and (iii) the stability in the execution times of tasks. Our analysis applies a five step procedure, called ProSim/ORP.

In an explorative study, we investigate a concrete ORP of small size. We perform three classes of parameter variations with a total of 600 simulation runs. Analyzing the results of this study, we observe that the impact of varying project planning parameters has statistically significant impact on project duration. Moreover, we observe high instability of developer assignments to tasks and strong variation of start and end times of feature development tasks in reaction to relatively small changes in estimates of developer productivity and feature-related effort.

INTRODUCTION AND MOTIVATION

Software product management addresses the problem of selecting and assigning features to a sequence of consecutive product releases. Operational product management studies the assignment of resources and their allocation to the tasks necessary to develop the features related to a single product release. The planning of a

single product release will be called Operational Release Planning (ORP) in the following.

ORP is known to be a cognitively and computationally difficult problem. In addition, some key parameters of the problem have to be estimated and are thus subject to uncertainty.

Uncertainty is an important issue in many software engineering planning problems. The uncertainty principle in software engineering was formulated by Ziv *et al.* as follows (Ziv *et al.*, 1996): “Uncertainty is inherent and inevitable in software development processes and products.” Uncertainty is also related to risk. As Kitchenham observes: “Senior managers and project managers need to concentrate more on managing estimate risk than looking for a magic solution to the estimation problem” (Kitchenham, 1998).

Uncertainty may arise from inaccurate or incomplete information, from linguistic imprecision and from disagreement between information sources. Uncertainty can be assumed in both informal knowledge (uncertainty due to lack of understanding or incomplete information) and in formalized knowledge (uncertainty due to imprecise information). The focus of this paper is to study the impact of the uncertainty due to imprecise information in the context of ORP.

Simulation in general has the ability to study the behavior of complex systems in greater detail. Both continuous and discrete process simulation approaches have the ability to study the sensitivity of current or past behavior of development projects due to uncertainty by means of systematic variation of model parameters (i.e., using Monte-Carlo type of analyses).

The structure of the paper is as follows. Section 2 describes related research on the topic of operational release planning and the issue of uncertainty in planning parameters related to this problem. Section 3 presents the goals of our research. Section 4 describes the method we applied to tackle the research goals. Section 5 describes an explorative case study and the results we achieved from applying our method. Section 6 discusses limitations and threats to validity. Finally, Section 7 presents conclusions and suggestions for follow-up research.

OPERATIONAL RELEASE PLANNING AND RELATED RESULTS

This Section introduces the problem of operational release planning and discusses related research.

The Problem of Operational Release Planning

We consider features to be "a logical unit of behavior that is specified by a set of functional and quality requirements" (van Gurp *et al.*, 2000)**Error! Reference source not found.** Features are an essential abstraction that both customers and developers understand. They provide an efficient way to manage the complexity and size of requirements.

We assume a set F of features $f(1), \dots, f(N)$ is decided to be developed in the next release. In ORP, the realization of each feature requires a sequence of individual tasks. For our purposes, the vector T consists of S different types of tasks $t(1), \dots, t(S)$. Task types correspond to the fundamental technical, managerial or support contributions necessary to develop software in general. Typically, these tasks cover at least design, implementation and different types of testing. Note that certain dependency relationships between task types can apply, e.g., testing cannot be started before some or all of the implementation has been finished. The possible dependency relationships are represented by a set called Dep .

The set of all possible feature/task(type)-combinations is denoted FT . It represents all individual feature/task(type)-combinations $ft(n,s)$, with $n = 1, \dots, N$, and $s = 1, \dots, S$.

Human resources (e.g., different types of developers, analysts, external collaborators) are intended to perform the different tasks needed to create the features. Our resource allocation process addresses the assignment of the individual human resources to tasks. Each developer is working on just one task at a time. An additional assumption is that for each task only one developer is allowed to work on it. In the case that different developers are allowed to work on the task, the original task would need to be further decomposed.

It is well known that there are strong differences in the skills and productivity of software developers (Acuña *et al.*, 2006). In order to accommodate this situation, we introduce an average skill level with a normalized productivity factor of 1.0 for each type of task. This allows consider more or less skilled developers having a higher or lower productivity factor than 1, respectively. We assume that the project manager can judge the different degrees of productivity of the developers.

We consider a pool D of M developers, each developer denoted by $d(1), \dots, d(M)$. Each developer can perform one or more types of development activities with a specific productivity. The productivity factor $p(i, j)$ of a developer $d(i)$ for performing a task of predefined type $t(j)$

indicates if the developer is able to perform the task at all ($p(i, j) \neq 0$), and if 'yes', how productive he/she is performing that task. We note that the assignment of a skill level for the different developers in dependence of a whole class of tasks is a simplification of reality as there might be further differentiation even within different areas of e.g. testing. While in principle the granularity of the definition of a task is flexible, we have to keep the model reasonable in size and consequently do not consider this advanced aspect.

ORP is defined as performing a set of tasks with the given resources in a way to minimize the overall release time. This release time is determined by the maximum end time of all feature/task-combinations. Each feature/task(type)-combination can be associated with an effort $eff(i, j)$, where i corresponds to feature $f(i)$ and j corresponds to task type $t(j)$. Effort $eff(i, j)$ then denotes the estimated effort needed to fulfill the feature-specific task of a specific type. The vector of all effort estimates is denoted E . Note that the duration needed to perform the feature/task(type)-combination depends not only on the workload, but also on the productivity of the developer assigned to this task. For example, if the effort for $ft(1,3)$ is estimated to be $eff(1,3) = 10$ person days, then developer $d(5)$ would need 5 days to perform the task $t(3)$ of feature $f(1)$, if the productivity of developer $d(5)$ for task type $t(3)$, e.g., testing, is assumed to be 2.

Each feature/task(type)-combination $ft(i, j)$ has a starting time $st(i, j)$ and an end time $et(i, j)$. The sets of all start times and all end times are denoted ST and ET , respectively.

Solving the ORP problem means to find an assignment of developers $d(k) \in D$ to all feature/task(type)-combinations $ft(i,j) \in FT$ such that the overall release time is minimized. More formally, the ORP problem is characterized by the following 9-tuple:

$\langle F, T, FT, D, E, P, Dep, ST, ET \rangle$

with

F – set of all features,

T – set of all task types

FT – set of all feature/task(type)-combinations,

D – set of all developers,

E – set of all estimated effort factors,

P – set of all estimated productivity factors,

Dep – precedence relationship between task types,

ST – set of starting times of all feature/task(type)-combinations, and

ET – set of end times of all feature/task(type)-combinations.

Related Research

Software project management includes management of human and other resources. Assignment of properly skilled resources to tasks is a complex process typically including a large number of variables related to schedule times, availability, staffing and skill profile. However,

intuition alone is not sufficient for making complex and crucial decisions. If this process is done ad hoc and manually, the results are poor resource utilization, schedule overrun and an overall fire-fighting mode to conduct projects.

In a recent Special Issue of IEEE Software, Pyster and Thayer stated that “Software Engineering Project Management’s progress has been agonizingly slow in many years, probably it is driven more by human behavior than by technology” (Pyster and Thayer, 2005). Similarly, Padberg pointed out that software engineering currently offers little help to software project managers on how to develop good schedules for their projects (Padberg, 2001).

There is a deficit on systematic approaches to provide operational guidance in software projects. Chang *et al.* proposed the application of genetic algorithms for software project management (Chang *et al.*, 2001). The proposed method just aimed at schedule minimization and was not looking into different productivity levels of developers. More recently, Fenton *et al.* used Bayesian belief networks as a means to circumvent the inherent uncertainty in effort estimates for resource decisions as part of software project management (Fenton *et al.*, 2004).

Strategic product management (Ruhe and Ngo-The, 2004) is responsible for mapping different features to different software releases whereas operational product management (Momoh, 2004) is responsible to answer how features selected for a single release will be developed. For this paper, we assume that the set of features to be released is fixed and the purpose of planning is to assign appropriate resources to the tasks. Ngo-The and Ruhe propose a method that considers both the planning of software releases and allocation of resources in individual releases simultaneously (Ruhe and Ngo-The, 2004). However, the impact of uncertainty has not been investigated.

Kellner *et al.* proposed several fundamental directions for applying process simulation in software engineering (Kellner *et al.*, 1999). All of these directions are in some way approaching the inherent uncertainty of software processes and related data. Control and operations management is one of them. Pfahl presents a five-step procedure called ProSim/RA that combines process simulation with Monte-Carlo simulation to analyze risk factors in software projects (Pfahl, 2005). ProSim/RA will be adapted in Section 4 for the purposes of the research presented in this paper.

RESEARCH OBJECTIVES

The main goal of this research is to investigate the impact of uncertainty in operational release planning. In more detail, we formulate three research questions and the related hypotheses. The research questions are formulated using a goal-oriented template (Johnson and Christensen, 2000):

Research Question 1	
Analyze (objects of interest)	A series of simulation runs with varying problem instances
In order to (purpose)	Understand the impact of uncertainty in productivity and/or effort estimates
With respect to (focus)	Release time
From the point of view of (perspective)	Project manager
For the environment (context)	Operational release planning project XYZ

Research Question 2	
Analyze (objects of interest)	A series of simulation runs with varying problem instances
In order to (purpose)	Understand the impact of uncertainty in productivity and/or effort estimates
With respect to (focus)	Stability of the assignment of human resources to tasks
From the point of view of (perspective)	Project manager
For the environment (context)	Operational release planning project XYZ

Research Question 3	
Analyze (objects of interest)	A series of simulation runs with varying problem instances
In order to (purpose)	Understand the impact of uncertainty in productivity and/or effort estimates
With respect to (focus)	Starting and ending times of tasks
From the point of view of (perspective)	Project manager
For the environment (context)	Operational release planning project XYZ

We formulate the research hypotheses in correspondence to the three research questions stated above. Each hypothesis is presented as a pair of *alternative hypothesis* and *null hypothesis*. The null hypothesis is directly tested; while the alternative hypothesis asserts the opposite of the null hypothesis. The alternative hypothesis is supported if the null hypothesis is rejected (Lott and Rombach, 1996).

The hypotheses corresponding to the three research questions are referred to as H_1 , H_2 , and H_3 , respectively. Their corresponding null hypotheses are labeled as H_{10} , H_{20} , and H_{30} . The alternative hypotheses are denoted by H_{11} , H_{21} , and H_{31} , respectively.

H_{11}	Decrease (increase) of productivity and increase (decrease) of effort results in significant increase (decrease) of project duration.
H_{10}	Decrease (increase) of productivity and increase (decrease) of effort does not result in significant increase (decrease) of project duration.

H_{21}	Decrease (increase) of productivity and increase (decrease) of effort results in significant instability in the assignment of human resources to tasks.
H_{20}	Decrease (increase) of productivity and increase (decrease) of effort does not result in significant instability in the assignment of human resources to tasks.

H_{31}	Decrease (increase) of productivity and increase (decrease) of effort results in significant instability in the start and end times of tasks.
H_{30}	Decrease (increase) of productivity and increase (decrease) of effort does not result in significant instability in the start and end times of tasks.

METHODOLOGY

To perform a systematic analysis of the impact of uncertain problem parameters in the ORP problem we adapt a risk analysis procedure called ProSim/RA (Process Simulation based Risk Analysis) for our purposes (Pfahl, 2005). The result is a five step procedure called ProSim/ORP that applies modeling, base-lining, simulation, and analysis of ORPs. For a given ORP problem $\langle F, T, FT, D, E, P, Dep, ST, ET \rangle$ the steps are:

- STEP 1: Determine baseline solution to the ORP problem.
- STEP 2: Identify ORP problem parameters and define uncertainty ranges.
- STEP 3: Define ORP performance parameters and set up simulations.
- STEP 4: Conduct Monte Carlo (MC) analyses.
- STEP 5: Analyze simulation results.

STEP 1 – Determine baseline solution to the ORP problem

The analysis of the impact of uncertainty is always done in relation to a baseline solution. In general, a solution of the ORP problem consists of the assignment of developers to feature/task-combinations such that the overall duration of the release is minimized under given constraints (i.e., available developers). To determine the baseline solution, we apply the process simulation model REPSIM-2, Release Plan Simulator Version-2 (Al-Emran and Pfahl, 2007). The resulting solution of the ORP problem is called the Baseline case (or solution).

The heuristic used in REPSIM-2 for assigning developers to feature/task-pairs essentially consists in matching the next available developer with the highest task-specific productivity to the next waiting feature with the largest effort (for a specific task). If only one developer with very low productivity is currently idle, then this mapping procedure can result in assigning a developer with low productivity to a large feature. To avoid such a worst case situation, a set of threshold variables are defined which exclude developers with productivity below a certain value to be assigned to feature/task-pairs. This threshold values for different tasks are defined by the simulation model itself whichever are the bests. According to the heuristic, a developer $d(k)$ will be assigned to a feature/task(type)-combination $ft(i, j)$ if the following conditions are full-filled:

- There is work still pending to be done for the feature/task(type)-combination $ft(i, j)$.
- The effort (work volume) $eff(i, j)$ of the waiting feature/task(type)-combination $ft(i, j)$ is the maximum of all waiting feature/task(type)-combinations.
- The productivity $p(k, j)$ of a candidate developer $d(k)$ related to a specific task type $t(j)$ is greater than the threshold productivity.
- A candidate developer with sufficient productivity is currently idle.
- If several equally large feature/task-combinations for the same feature but different tasks are subject to be assigned the same developer at the same point in time, the predecessor task will be worked on first.

STEP 2 – Identify ORP problem parameters and define uncertainty ranges

Uncertainty in the project data can be related to various problem parameters of the ORP problem. Typically, the feature/task-specific effort estimates, the estimated number and productivities of developers, dependencies between features or task, and other factors are candidates to be selected in this step. Uncertainty ranges can be based on observed variation of estimated data in past projects or on expert opinion. In the explorative case study presented in Section 5 below, we use triangular distributions to describe uncertainty ranges of selected ORP problem parameters.

STEP 3 – Define ORP performance parameters and set up simulations

The performance parameters of the ORP which are supposed to be affected by the variation of the problem parameters are defined in this step. In addition, planning is performed on the number and the type of simulation runs to be performed. Typical performance parameters are duration of the release, effort consumption, idle times of developers, start and end times of feature-specific tasks, etc. In the explorative case study presented in Section 5 below, we focus on release duration and several parameters that describe the structure of an ORP.

STEP 4 – Conduct Monte Carlo analyses

In this step, using the process simulation model REPSIM-2, the simulations defined in the previous steps are conducted. Running MC analyses means systematic sampling of problem parameters from the triangle distributions defined in STEP 2.

4.5 STEP 5 – Analysis of results

The results from simulation runs conducted in the previous step are analyzed. The analysis usually involves summary and descriptive statistics (mean, standard deviation, scatter plots), as well as inference statistics involving formal hypothesis testing. In the explorative case study presented in Section 5 below, we analyze the hypotheses defined in Section 3.

CASE STUDY

We are currently applying ProSim/ORP to several release planning cases, including cases with more than 60 features and more than 12 developers. Due to space limitations, we present only one case here.

Baseline solution to the ORP problem

In order to investigate the research questions stated in Section 3, we conducted a small scale explorative case study for illustrative purposes. Similar investigations for larger problems are ongoing.

The ORP problem studied involves the following problem parameters:

- 8 features to be developed: $f(1), \dots, f(8)$.
- 3 task types $t(1), \dots, t(3)$, e.g., design, implementation, and test, respectively.
- 6 developers available to work on each feature/task(type)-combination: $d(1), \dots, d(6)$.
- The estimated work volume (parameter eff , measured in person-weeks) for each feature/task(type)-combination, and the estimated productivity (parameter p , dimensionless) of each developer per task type are as of Tables 1a and 1b, respectively.

Figure 1 shows the ORP generated by REPSIM-2 using the parameter settings shown in Table 1. For the

following steps, this plan will be referred to as the Baseline case. The total duration of the release is about 22.7 weeks. Note that a 100% dependency among the tasks of the same feature is considered for this study. That means, a task T_i of a feature can start if and only if task T_{i-1} of the same feature is completed. However, this case is not a fixed configuration and REPSIM-2 is able to consider any task dependency from 0% to 100%.

Identify ORP problem parameters and define uncertainty ranges

In our case study we focus on the impact of uncertainty in ORP problem parameters effort (E) and productivity (P). Table 2 shows the variation of risk factors in terms of percentage with respect to their original value shown in Table 1.

Each of the 12 cases shown in Table 2 represents different probability distribution sampling patterns for effort and productivity estimates. Both Effort and Productivity parameter values are sampled from triangle distributions of the type TRIANG (min, peak, max). Here, peak represents the most probable value: values shown in Table 1 (assuming estimations are correct); min represents the minimum value: values obtained from Table 1 and then by multiplying them with factors shown in “Min” column of Table 2 (assuming values are over-estimated); max represents the maximum value: values obtained from Table 1 and then by multiplying them with factors shown in “Max” column of Table 2 (assuming values are under-estimated). For example, in Case 1 the sampling distribution for $eff(1, 1)$ is TRIANG (2.7, 3, 3.6); in Case 5 the sampling distribution for $p(4, 2)$ is TRIANG (1.6, 2, 2.2); in Case 11 the sampling distribution for $eff(8, 3)$ and $p(6, 1)$ are TRIANG (8, 10, 14) and TRIANG (0.6, 1, 1.2), respectively.

In runs 1 and 3 asymmetric triangle distributions were applied. These cases represent situations that reflect systematic underestimation of effort in the Baseline case. The sampling from asymmetric triangle distributions in cases 5 and 7 represents situations that reflect systematic overestimation of productivity in the Baseline case. Finally, the asymmetric cases 9 and 11 reflect situations in which in the Baseline case both effort and productivity were systematically underestimated respectively overestimated.

Define ORP performance parameters and set up simulations.

The following ORP performance parameters were selected for this study:

- Release duration [weeks]: represents the make span of a release in calendar weeks from the beginning of a release construction to the end of its realization. Due to the variation in risk factors, the duration of a release may change. The variable representing a run-

specific duration is termed Dur_run. It's dimension is weeks.

- Developer assignment [no unit]: represents allocation of developer of an operational plan. Re-allocation may need to be done when performing re-planning due to the variation in risk factors. The number of such re-allocation needed with respect to the baseline case is termed Alloc_diff.
- Feature/task scheduling [no unit]: represents the differences between start times of feature-specific tasks (measure ST_diff) and the differences between end times of feature-specific tasks (measure ET_diff). Based on the values of ST_diff and ET_diff we calculated the difference between feature-specific task execution times (measure Dv_diff) using the following formula:

$$Dv_diff = \frac{d_1 + d_2}{d_1 + d_2 + d_3}, \text{ with}$$

$$d_1 = \sqrt{\sum_{j=1}^{24} (ST_j - ST_{baseline})^2},$$

$$d_2 = \sqrt{\sum_{j=1}^{24} (ET_j - ET_{baseline})^2},$$

$$d_3 = \sqrt{\sum_{j=1}^{24} overlap_j^2},$$

where overlap_i refers to the duration of overlap between a feature-specific task in the i-th simulation run and the Baseline case. If there is no overlap, then Dv_diff = 1, if the durations of two feature-specific tasks are identical and have an overlap of 100%, then Dv_diff = 0.

The simulation runs were set up using the triangle distributions presented in Table 2. For each of the 12 cases, 50 simulations were planned to be run.

Conduct simulation-based MC analyses

Running 50 simulations with REPSIM-2 for each of the 12 cases defined in STEP 2 produced the results shown in Tables 3a-c below.

Table 3a shows the means and standard deviations of performance variable duration. The values in the column "mean" represent the means of measure Dur_run defined in Section 5.2.

Table 3b shows the means and standard deviations of performance variable developer allocation. The values in the column "mean" represent the means of measures Alloc_diff defined in Section 5.2.

Table 3c shows the means and standard deviations of performance variable feature/task scheduling. The values in the columns denoted "mean" represent the means of variables ST_diff, ET_diff, and Dv_diff, respectively, defined in Section 5.2.

Due to space limitations it is not possible to show details on the descriptive statistics on all of the cases. However, to provide better understanding on how the empirical distributions of Alloc_diff, ST_diff, ET_diff and Dv_diff look like, we provide in Figure 2 the data related to case 10.

Analyzing Simulation Results

The goal of the case study was to investigate the hypotheses presented in Section 3. The first hypothesis was related to release duration:

H ₁₁	Decrease (increase) of productivity and increase (decrease) of effort results in significant increase (decrease) of project duration.
H ₁₀	Decrease (increase) of productivity and increase (decrease) of effort does not result in significant increase (decrease) of project duration.

In order to check whether average release duration is significantly different from the duration in the baseline case, we first had a look at the summary statistics shown in Table 4. It is not surprising to see that the duration on average increases in the cases 1, 3, 5, 7, 9, and 11, as these cases used asymmetric triangle distributions for sampling effort and/or productivity data. Since the asymmetry is positive in the case of effort and negative in the case of productivity, it could be expected that duration will increase on average.

Also in the symmetric cases (2, 4, 6, 8, 10, 12) a slight increase in duration can be observed. In order to check whether the differences are significant, we applied a single sample t-test. It turned out that at an alpha level of 5% in all cases but cases 2 and 10 the average duration was significantly larger than the duration of 22.71875 weeks in the Baseline case.

To make sure that the t-test is applicable, we applied a Kolmogoroff-Smirnoff test to check for normality of the simulation data. As data in Table 6 shows, in all cases but case 10, normality can be assumed, and even in case 10 the p-Value is very close to 0.05, implying that the assumption of non-normality could almost have been rejected.

Although the results presented in Table 5 indicate that uncertainty in estimates of important parameters as feature effort and developer productivity seems bear the risk of schedule overruns, the size of duration increase is less frightening than one might expect. Even in the case of most extreme variation of effort and productivity (case 11) average duration excess is only about 16%, a number that is small compared to project duration overruns otherwise published in the literature.

Thus, we decided to look deeper into the issue. We assumed that one reason that might account for the relative weak (negative) effects on release duration is the possibility of compensating overestimation or underestimation of efforts and productivities by re-allocating developers to feature/tasks. In order to analyze the issue, we checked for structural changes in the release plans generated by REPSIM-2.

As described in Section 5.3, structural changes were measured in two different ways. Firstly, we counted the number of changes in allocations of developers to feature/task-combinations (measure Alloc_diff). Then we measured the differences between start times of feature-specific tasks (measure ST_diff) and the differences between end times of feature-specific tasks (measure ET_diff). Based on the values of ST_diff and ET_diff we calculated the difference between feature-specific task execution times (measure Dv_diff). The analyses conducted on these data relate to hypotheses H₂ and H₃.

H ₂₁	Decrease (increase) of productivity and increase (decrease) of effort results in significant instability in the assignment of human resources to tasks.
H ₂₀	Increase (decrease) of productivity and increase (decrease) of effort does not result in significant instability in the assignment of human resources to tasks.

H ₃₁	Decrease (increase) of productivity and increase (decrease) of effort results in significant instability in the start and end times of tasks.
H ₃₀	Decrease (increase) of productivity and increase (decrease) of effort does not result in significant instability in the start and end times of tasks.

Figure 2 shows the results of the measurements for case 10. When looking at the data of measure Alloc_diff (see also Table 3b), one can see that the number of different developer allocations between a simulation run and the Baseline case is on average larger than 12. Since the total number of allocations is 24 (8 features with 3 associated tasks each), this number represents a change in developer allocation of more than 50% on average.

The impact of effort and productivity variation of +/- 20% is even stronger when looking at measure Dv_diff. One can see that the distance between the schedules of feature-specific tasks in the simulations run i as compared to the Baseline case is on average more than 60%. Thus, both measures, Alloc_diff and Dv_diff, indicate a massive influence on developer allocation and scheduling of feature-specific tasks.

Tables 3b and 3c summarize the structural changes for all 12 cases. One can observe that even in cases with only 20% variation of effort *or* productivity, substantial re-allocations of developers have to be applied in order to keep the overall duration under control. Similarly strong structural changes in the task scheduling can be derived from measures ST_diff, ET_diff and Dv_diff. In conclusion, one may say that relatively small changes in duration (which are not even significant for case 10) seem to be complemented by massive structural changes in operational release plans. This is certainly an interesting result as it demonstrates the high sensitivity of operational release plan structures to uncertainty in effort and productivity estimates.

THREATS AND LIMITATIONS

The proposed methodology is based on knowledge and experience from domains such as project management, Monte-Carlo simulation and process simulation. In a strong sense, we have reported in detail the results from one operational planning trial example based on this methodology. We are currently running simulations on larger case examples and we haven't yet observed fundamentally different results.

All the conclusions drawn for the stated trial are based on a large set of 50 randomly generated test examples in each of the classes of parameter variations. That means that for the test example the treatment-outcome relationship a statistically valid for the three research questions raised.

A fundamental limitation of any type of Monte-Carlo simulation is the validity of the assumed data distribution. The triangular function is one of the more frequently applied distributions in real-world which has often proven to be a sufficiently good abstraction of the real-world situation. We plan to validate the stated hypotheses for other distributions as well.

Internal validity is asking for the treatment-outcome construct which is automatically fulfilled in this type of investigations. Other validity questions are related to construct validity. We assumed changes in the productivity and in the effort estimates and studied a combination of these two types of changes. Other uncertainties might occur as well such as the ones related to dependencies between tasks. For the outcome description, the release time is often used in project (release) planning. The other measures are newly defined and need complementary empirical validation.

All the determined solutions to the ORP problem were obtained from running a heuristic-based process simulation. While the heuristic does not guarantee optimal results, it was shown by a comparative analysis with the application of optimization that the results of this special heuristic are sufficiently good on average (Al-Emran, 2006).

Despite the stated limitation, we consider the achieved results an encouraging starting point to further investigate and explain the impact of different types of uncertainty on the results of operational release planning. It is too early to claim external validity, but it is fair to say that there is at least a better understanding of the nature of impact. In addition, we have shown the proposed methodology being applicable to conduct this and also some follow-up investigations being of the same type.

SUMMARY AND CONCLUSIONS

In this paper, we presented results of an exploratory study that investigated the impact of uncertainty related to developer productivity and feature/task effort data on operational release plans. More precisely, we studied the impact of these changes on (i) the duration of a release (ii) the stability of the assignment of human resources to tasks, and (iii) the stability in the execution times of the tasks. The results of the study indicate that even if duration of operational release plans is not as much affected by uncertainty in productivity and effort estimates, this relatively positive aspect for practitioners is compromised by the finding that control over duration is only achieved through massive structural changes in the operational release plan. Changes affect both allocation of developers to feature/task-combinations as well as the scheduling of feature specific tasks. Estimation uncertainty in the range of maximal $\pm 20\%$ can require to re-assign more than 50% of all developer allocations to feature/task-combinations, if duration shall be kept under control.

As a next step, it might be interesting to test how much the duration of operational release plans is affected, if stability in the structure of release plans is enforced. This could be done either by fixing both the allocation of developers to feature/task-combinations and by keeping the schedules for feature-specific tasks fixed. Such an analysis has already been conducted (Pfahl *et al.*, 2005). The results show that variation in effort and productivity can result in large work backlogs, i.e., work cannot be finished if no changes in developer allocation and task scheduling are allowed.

An even more interesting analysis, however, would be to investigate the impact on release plan duration if the allocation of developers to feature-specific tasks is fixed, but a re-scheduling of tasks is allowed. This situation would allow the release to be completed. How much the duration is affected on average is affected can only be speculated. Our assumption is that duration will be on average significantly longer than the average durations presented in this paper (Table 3a). This type of analysis will be one of our next research goals.

ACKNOWLEDGEMENTS

Part of the work presented was financially supported by the Natural Sciences and Engineering Research Council (NSERC) of Canada under Discovery Grant no. 327665-06, and by the Informatics Circle of Research Excellence (iCORE) of Alberta in Canada.

REFERENCES

- S. Acuña, N. Juristo, and A.M. Moreno, "Emphasizing Human Capabilities in Software Development", *IEEE Software*, Vol. 23, No. 2, 2006, pp. 94-101.
- A. Al-Emran, "Dynamic Re-Planning of Software Releases", Master's Thesis, University of Calgary, 2006.
- A. Al-Emran, and D. Pfahl, "Operational Planning, Re-Planning and Risk Analysis for Software Releases". (accepted for publication in the proceedings of PROFES 2007).
- C. Chang, M. Christensen, and T. Zhang, "Genetic Algorithms for Project Management", *Annals of Software Engineering*, Vol. 11, 2001, pp. 107-139.
- N. Fenton, W. Marsh, M. Neil, P. Cates, S. Forey, and T. Tailor, "Making Resource Decisions for Software Projects", *26th International Conference on Software Engineering (ICSE 2004)*, May 2004, pp. 397-406.
- J. van Gurp, J. Bosch, J., and M. Svahnberg, "Managing Variability in Software Product Lines", in *Proceedings of LAC'2000*, Amsterdam.
- B. Johnson, L. Christensen, "Educational Research-Quantitative and Qualitative Approaches", Allyn and Bacon, USA, 2000.
- A. Ngo-The, and G. Ruhe, "Optimized Resource Allocation for Incremental Software Development", *TR 062/2006, Laboratory for Software Engineering Decision Support, University of Calgary*, 2006.
- M.I. Kellner, R.J. Madachy, D.M. Raffo, "Software Process Simulation Modeling: Why? What? How?", *Journal of Systems and Software*, Vol. 46, 1999, pp. 91-105.
- B.A. Kitchenham, "The Certainty of Uncertainty", *European Software Measurement Conference*, Proceedings of FEMSA 98, Antwerp, Technologisch Institute, 1998.
- C. Lott, H. Rombach, "Repeatable Software Engineering Experiments for Comparing Defect-Detection Techniques", *Empirical Software Engineering*, Vol. 1, No. 3, 1996, pp. 241-277.
- J. Momoh, "Applying Intelligent Decision Support to Determine Operational Feasibility of Strategic Software Release Planning", Master's Thesis, Department of Electrical and Computer Engineering, University of Calgary, Canada, 2004.
- F. Padberg, "Scheduling software projects to minimize the development time and cost with a given staff", *8th Asia-Pacific Software Engineering Conference (APSEC 2001)*, 2001, pp. 187 - 194.

D. Pfahl, “ProSim/RA – Software Process Simulation in Support of Risk Assessment”, *Value-based Software Engineering*, Biffel S. et al. (eds.), Berlin: Springer, 2005, pp. 263-286.

D. Pfahl., A. Al-Emran, and G. Ruhe, “Simulation-Based Stability Analysis for Software Release Plans”, *International Software Process Workshop and International Workshop on Software Process Simulation and Modeling, SPW/ProSim 2006 - Proceedings*, Lecture Notes in Computer Science, 3966, Wang Q, Pfahl D, Raffo D, Wernick P (eds), Springer-Verlag: Berlin-Heidelberg, 2006, pp. 262-273.

A.B. Pyster, and R.H. Thayer, “Guest Editors' Introduction: Software Engineering Project Management 20 Years Later”, *IEEE Software*, Vol. 22, No. 5, 2005, pp. 18-21.

G. Ruhe, and A. Ngo-The, “Hybrid Intelligence in Software Release Planning”, *International Journal of Hybrid Intelligent Systems*, Vol. 1, No. 2, 2004, pp. 99-110.

H. Ziv, D.J. Richardson, and R. Klösch, “The Uncertainty Principle in Software Engineering”, *Technical Report UCI-TR-96-33*, University of California, Irvine, August 1996.

FIGURES AND TABLES

Table 1a. Feature/Task-specific efforts

		eff(i, j) [person-week]							
		f(1)	f(2)	f(3)	f(4)	f(5)	f(6)	f(7)	f(8)
Task	t(1)	3	8	6	3	5	7	10	6
	t(2)	6	3	10	3	6	5	5	8
	t(3)	6	2	5	6	4	3	6	10

Table 1b. Developer-specific productivities

		p(k, j) [no unit]					
		d(1)	d(2)	d(3)	d(4)	d(5)	d(6)
Task	t(1)	1.5	1	2	0	0.5	2
	t(2)	2	1.5	1	2	1.5	1
	t(3)	1	2	0	1.5	2	1

Table 2. ProSim/ORP risk factor variation

Case	Variation Parameter	Min	Max
Case 1	Effort	-10%	20%
Case 2	Effort	-20%	20%
Case 3	Effort	-20%	40%
Case 4	Effort	-40%	40%
Case 5	Productivity	-20%	10%
Case 6	Productivity	-20%	20%
Case 7	Productivity	-40%	20%
Case 8	Productivity	-40%	40%
Case 9	Mixed	Case 1 + Case 5	
Case 10	Mixed	Case 2 + Case 6	
Case 11	Mixed	Case 3 + Case 7	
Case 12	Mixed	Case 4 + Case 8	

Table 3a. Summary statistics of ORP performance parameter Dur_run

Case	Baseline Duration [weeks]	Run-specific duration Dur_run [weeks]	
		Mean	Standard dev.
Case 1	22.71875	23.811	1.611
Case 2	22.71875	23.146	1.608
Case 3	22.71875	24.755	2.210
Case 4	22.71875	23.526	2.428
Case 5	22.71875	23.780	1.012
Case 6	22.71875	23.075	1.250
Case 7	22.71875	24.849	2.034
Case 8	22.71875	23.508	2.141
Case 9	22.71875	24.541	1.434
Case 10	22.71875	23.038	2.080
Case 11	22.71875	26.367	2.495
Case 12	22.71875	23.950	3.331

Table 3b. Summary statistics of ORP performance parameter Alloc_diff

Case	Difference in developer allocation Alloc_diff [no unit]	
	Mean	Standard dev.
Case 1	7.2	3.289
Case 2	9.04	3.194
Case 3	9.765	2.688
Case 4	10.14	2.298
Case 5	11.38	3.562
Case 6	12.26	2.776
Case 7	13.3	2.644
Case 8	14.06	3.191
Case 9	11.96	3.201
Case 10	12.16	2.78
Case 11	13.08	2.806
Case 12	13.3	2.652

Table 3c. Summary statistics of ORP performance parameters ST_diff, ET_diff, Dv_diff

Case	ST_diff		ET_diff		Dv_diff	
	Mean	Std. Dev.	Mean	Std. Dev.	Mean	Std. Dev.
Case 1	19.26	11.34	25.44	11.9	0.4214	0.1503
Case 2	27.67	16.67	35.28	17.79	0.5026	0.1533
Case 3	32.72	17.02	42.93	17.76	0.5345	0.136
Case 4	40.76	17.8	52.43	19.52	0.6002	0.1299
Case 5	28.15	17.8	33.71	17.84	0.4958	0.1465
Case 6	34.28	17.55	40.8	17.48	0.5536	0.1344
Case 7	46.01	20.96	54.09	21.32	0.6096	0.1306
Case 8	47.95	20.49	57.06	21.55	0.6368	0.112
Case 9	38.29	18.62	47.41	19.19	0.5742	0.123
Case 10	41.05	16.03	49.71	16.26	0.61	0.0987
Case 11	60.34	22.13	75.17	26.19	0.6902	0.0817
Case 12	62.53	32.56	75.81	30.18	0.7052	0.0717

Table 4. Average differences of problem and performance parameters compared to the baseline case

Case (50 runs each)	Deviation of Effort Mean from Baseline Effort	Deviation of Productivity Mean from Baseline Productivity	Deviation of Duration Mean from Baseline Duration
Case 1	3.4%	0%	4.8%
Case 2	0%	0%	1.9%
Case 3	6.8%	0%	9.0%
Case 4	0%	0%	3.6%
Case 5	0%	-3.3%	4.7%
Case 6	0%	0%	1.6%
Case 7	0%	-6.7%	9.4%
Case 8	0%	0%	3.5%
Case 9	3.4%	-3.7%	8.0%
Case 10	0%	0%	1.4%
Case 11	6.9%	-7.5%	16.1%
Case 12	0%	-0.1%	5.4%

Table 5. Results from the single sample t-tests comparing the distribution of the average durations in each case with the duration in the baseline case

Case	Baseline Duration	Duration Mean	Duration Std. Dev.	p-Value
Case 1	22.71875	23.811	1.611	0.000
Case 2	22.71875	23.146	1.608	0.066
Case 3	22.71875	24.755	2.210	0.000
Case 4	22.71875	23.526	2.428	0.023
Case 5	22.71875	23.780	1.012	0.000
Case 6	22.71875	23.075	1.250	0.049
Case 7	22.71875	24.849	2.034	0.000
Case 8	22.71875	23.508	2.141	0.012
Case 9	22.71875	24.541	1.434	0.000
Case 10	22.71875	23.038	2.080	0.284
Case 11	22.71875	26.367	2.495	0.000
Case 12	22.71875	23.950	3.331	0.012

Table 6. Results of the Kolmogorov-Smirnov Normality test for duration with a Confidence Interval of 95%

Case	Duration Mean	p-Value
Case 1	23.811	0.111
Case 2	23.146	>0.150
Case 3	24.755	0.121
Case 4	23.526	>0.150
Case 5	23.780	0.103
Case 6	23.075	>0.150
Case 7	24.849	>0.150
Case 8	23.508	>0.150
Case 9	24.541	>0.150
Case 10	23.038	0.049
Case 11	26.367	>0.150
Case 12	23.950	>0.150

Assignment of developers to feature/task-combinations:

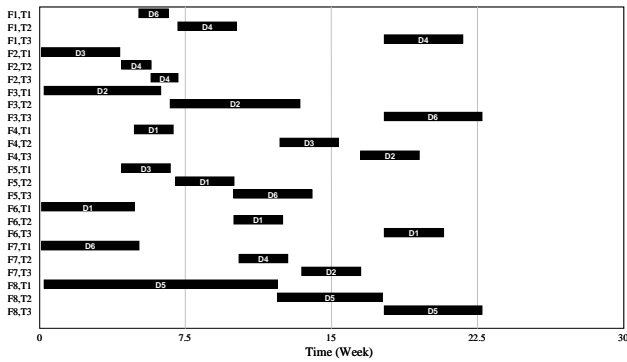


Fig. 1. The operational release plan of the Baseline Case

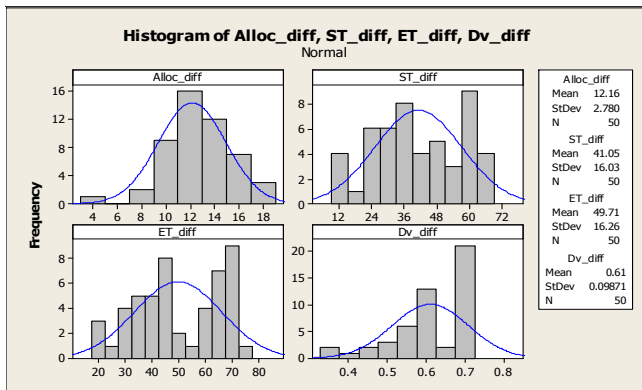


Fig. 2. Example descriptive statistics of ORP performance variables Alloc_diff, ST_diff, ET_diff and Dv_diff (Case 10)