Quantifying fair payment after outsourcing – a case study

Laurenz Eveleens, Peter Kampstra*,[†] and Chris Verhoef

VU University Amsterdam, Department of Computer Science, 1081 HV, Amsterdam, The Netherlands

ABSTRACT

After outsourcing, issues can arise between the outsourcing organization and their third parties about fair payment. In a *ceteris paribus* situation, fair payment can be determined based on the differences made by the third parties. However, when information technology development is (partly) outsourced to third parties, often, not only does the supplier change, but the development process is also changed. This change of development process alone influences important software metrics, such as time to market, productivity, costs, size, and quality, exactly the metrics that are often used to establish payments. Quantifying the influence of the development process redesign is therefore vital to make a fair assessment of the changes truly caused by the supplier, and thus realistic payment.

In this paper, the influence of the change in the development process on important key performance indicators is quantified using simulation techniques. We use discrete event simulation to analyze the effect on time to market. The techniques we use are illustrated and validated by applying them in a real-world situation. The techniques can be used to estimate the influence of the business process change both before outsourcing and after the outsourcing has been decided. Our case study helped the organization in their outsourcing by adapting its proposed development process so that the balance between a more formal process and time to market became more in line with their demands. Although the specific numbers will be different per company, other organizations can apply the general principle. Copyright © 2015 John Wiley & Sons, Ltd.

Received 17 October 2014; Accepted 13 November 2014

KEY WORDS: outsourcing; time to market; simulation; software development process; development process; redesign; information technology metrics

1. INTRODUCTION

According to companies like Gartner or Plunkett Research [26, 22], the global annual information technology (IT) outsourcing volume has grown to hundreds of billions of dollars. Indeed, there are many reasons to, partly or completely, outsource IT development to third parties. Important reasons for outsourcing are cost reduction, productivity increase, availability of personnel, quality improvement, and time to market. Often, outsourcing contracts contain targets related to these needs: cost reduction targets, service level agreements (SLAs) for response times, quality levels, and productivity improvement.

To enable an outsourcing deal, many processes usually have to be changed. Often, because of a lack of trust or trustworthiness, more checks and balances are built into the processes not only by the outsourcer but also by the third party to prevent problems during outsourcing. For example, in a study about lack of trust in global software development, the quality management system for developing software comprised of 'more than 500 documents and around 100 process descriptions'

^{*}Correspondence to: Peter Kampstra, Department of Computer Science, VU University Amsterdam, De Boelelaan 1081a, 1081 HV Amsterdam, The Netherlands.

[†]E-mail: pkampst@few.vu.nl

[21]. This easily leads to performance being taken over by conformance. Depending on the situation, we go from trust me to tell me, to show me, and to prove me.

The changes in processes cause the outsourcing transition to affect the software metrics in two ways. First, they are influenced by changes made in the development process. Second, they are affected by the change of supplier. These two changes do not have to be in the same direction. For example, a new process can negatively affect productivity, while a different supplier positively influences the productivity. Reverse movements, for one or both of these changes, are possible as well. For example, a new process can impact the productivity positively, while a supplier can have a negative effect on productivity.

Such changes in processes have an impact on the key performance indicators (KPIs) that are used in domains such as cost, size, quality, time to market, and productivity. For example, a more formal process may introduce more approvals, more documentation, and more rigid change request policies. These additions in turn potentially increase the time to market. Other KPIs are likely affected as well. In order to properly assess the KPIs, it is necessary to baseline them before the outsourcing transition and measure them after transition, as should be performed in any business process redesign setting [19].

Often, a baseline *before outsourcing* of KPIs is taken as the starting point of a payment model. Then, per contract, shorter time to market, higher productivity, and/or higher quality is agreed upon. However, a fair assessment of the newly achieved values for the KPIs needs to take into account not only the suppliers' influence but also the influence of the changing process and their potential reverse movements. That is why, it is important to investigate these effects to distinguish the true impact of the third party.

In this paper, we focus on these two effects on IT metrics after an outsourcing transition. We quantify the changes in IT metrics due to the changes in the development process to fairly assess the impact of the supplier on them. We illustrate our approach with a real-world case of an organization that was in the middle of a transition to outsource a multimillion dollar IT portfolio in a multiparty context.

Normally, contracts are signed before any work is carried out, so at the time of agreeing on quantitative targets, information as illustrated earlier cannot be directly available. This makes it difficult for the offering organization and the insourcer to agree on those targets. We propose to estimate the transition effect on the time to market by means of a simulation. We simulated the approval durations and compared them with the approval durations that were measured before the transition to quantify their net effects on the newly proposed process. The result of our simulations was that the process was adjusted to cater for more trust and less control. The initially designed process contained too much control so that time to market would increase to unacceptable levels.

Simulation is a well-known technique that can be used to show the possible real effects of alternative conditions and courses of action. Many other papers have been published during the last decades about theoretical and practical software process simulation modeling (SPSM) [17, 2, 1]. A systematic literature review was published in 2008 by Zhang *et al.* [37]. They analyzed 209 papers and distinguished four types of SPSM studies:

- (A) software process simulation model or simulators;
- (B) process simulation modeling paradigms, methodologies, and environments;
- (C) applications, guidelines, and frameworks of adopting process simulation in software engineering practice; and
- (D) experience reports of SPSM research and practice.

According to this classification, our paper can be characterized as B and D. As recommended by Sutton [32], the focus of this paper is not just on modeling and simulation but also on a novel way of analyzing an intended software process to be used after an outsourcing transition.

The numbers that are given in the rest of this paper have been indexed, rounded, or otherwise adjusted for confidentiality reasons. The tools we used for the analyses include R (a variant of the statistical program S-plus) [23], Arena [18] (a discrete event simulator), Business Objects [27] (for retrieving management data), and Microsoft Excel with Crystal Ball [9] (for Monte Carlo simulation). Any other combination of tools with similar functionality would have been as appropriate.

1.1. Organization

The remainder of the paper is divided into multiple sections. Section 2 explains the two effects after a transition in more detail. Moreover, it shows why simulation is needed to estimate a single effect. Section 3 describes the case study and the software development process used in the organization. After scrutinizing the available data in more detail, it turned out that there is overlap of process steps. Section 4 shows how the proposed development process model is used to create a simulation model of days spent waiting for approval. We validate our simulation model by using data from the process that was implemented, an altered version of the proposed process. Section 5 discusses the results of the simulation and uses the findings from the process before transition to show the meaning of the results. Our simulation focuses on time to market, but we also discuss other IT metrics in Section 6. Section 7 illustrates other useful applications of software process simulation, and in Section 8, we discuss the use of simulation and IT metrics in general. Section 9 concludes the findings.

2. TWO EFFECTS AFTER TRANSITION

In the introduction, we stated that important KPIs are affected in two ways by a transition. First, they are influenced by changes made in the development process. Second, they are affected by the change of supplier. We will explain these two effects with two examples, one with productivity and one with time to market.

Suppose that after the transition, the organization detects that IT productivity has decreased and blames the insourcer (vendor). An insourcer may claim it has achieved its target of productivity and is not to blame for the decrease. In fact, it is possible that both statements are true. Even though the insourcer has a higher productivity, after the transition, it is possible that the overall productivity decreases because of a change of process.

In Figure 1, the two effects of a transition are shown in a more systematic way. On the one hand, there is the new process, which usually tends to decrease productivity because more formalities are put in place, which take time but do not always produce extra value. On the other hand, there is the hopefully higher productivity of the insourcer (supplier) whose core competence is IT. In this paper, we discuss ways to quantify the effect of the transition, that is, the difference between states 1 and 3 or states 2 and 4. Given the overall productivity change, this quantification enables us to, for instance, calculate the part that is accountable to the vendor, that is, the difference between states 1 and 2 or states 3 and 4.

Another example is time to market. The time to market is also affected by a transition in two ways. For example, if we assume a transition to another more formal process, it is expected to



Figure 1. Two effects after an outsourcing transition on an information technology metric, here, for productivity (one effect per axis).

increase. On the other hand, expertise, overtime, and other factors introduced by the vendor influence time to market positively.

After a transition has been completed, it is possible to measure the effect of the transition. We show this with the following example. Suppose the approval duration has increased by an average of 5 days per project during which no work is performed, and suppose the average duration of an IT project constructed via the old process was 200 days, while the size of the projects remains to be about the same under the new regime. This means that the transition effect has caused a 5/200 = 2.5% increase of the duration. In this case, the time to market should be corrected by a factor of 1.025 to compensate for the new process with more waiting times in order to compare it with the old situation. The 1.025 factor compensates for the process change, so after measuring the total time to market, this factor is used to calculate the effect of the supplier. This corrected time to market is then used to assess whether the targets in the contracts are met. In fact, what we do is account for approval duration, because that is the major impact of process changes. And it is up to the supplier to meet the targets in between waiting.

Normally, contracts are signed before any work is carried out, so at the time of agreeing on quantitative targets, information as illustrated earlier cannot be directly available. This makes it difficult for the offering organization and the insourcer to agree on those targets. We propose to estimate the transition effect on the productivity by means of a simulation.

If the setup of the new and more formal process is known, this allows a simulation model to be made of the process. We will do this in a case study. By calculating thousands of simulated projects, it gives an estimate of the increased approval duration, providing valuable information to support the outsourcing decision making.

3. CASE STUDY

Some organizations shape their sourcing deals with multiparty master agreements, where, for instance, different parties take care of different steps in the software life cycle. For example, requirements are engineered by the organization itself, implementation is carried out by the development partner, testing is performed by the testing partner, and deployment is carried out by yet another partner. This is sometimes called multisourcing [12].

Our case study involves a large financial organization. This organization was in the middle of a transition to outsource their multimillion dollar IT portfolio in a multisourcing context. A transition was being made from an in-house development setting to a multiparty outsourcing process. The development, which was being performed in the so-called design-and-build phase, was to be outsourced to another country.

To accomplish this, a new development process was proposed, which contained more approvals. As stated in the introduction, because of the new development process, KPIs are affected in two ways: by the new development process and the change in supplier. In order to make a fair assessment of the supplier, first, the impact of the new development process needed to be estimated. To quantify this impact, we need both the historic KPIs and their estimates in the new development process.

This section focuses on the historic situation. First, for completeness' sake, the historic development process, the Dynamic Systems Development Method (DSDM), is described. Those familiar with this process can disregard this subsection.

After that, a historic KPI is quantified: the average duration of IT projects. This is performed by investigating the durations of 318 projects that have been executed. The various durations of the various phases allow us to determine the historic benchmark needed to quantify the impact of the new development process, which we will show in Section 4.

3.1. Historical development process

In our industrial case study, a customized variant of the DSDM is used. Information on the DSDM is available in various sources [30, 29, 28]. The DSDM method consists of multiple phases, and during each phase, certain predefined activities take place. The following phases are used in this organization.

- Feasibility study (FS). During this phase, the project's feasibility is determined in terms of technical realization, costs, duration, risks involved, and so on.
- Business study (BS). The BS follows the FS. This phase determines things like the context of the project, requirements, and system architecture.
- Functional model iteration (FMI). In this phase, further (non)functional requirements and information are being gathered, usually by prototyping. This phase is repeated if necessary.
- Design-and-build iteration (DBI). In this phase, the actual system is being developed so that the (non)functional requirements are met. This phase is also iterated when necessary.
- Acceptance tests. While not standard for the DSDM method, this organization added a formal test and acceptance phase. During acceptance, a third party tests the delivered system for compliance to regulations and existing systems. The tests are split into system testing (ST), functional acceptance (FA), and production acceptance (PA).
- Implementation (IMPL). During IMPL, the system is being delivered and evaluated.

There are three levels of detail that are relevant for a simulation of the development process. We explain them.

- Project level. The project level concerns the already discussed phases of the customized DSDM process. It is the top-most level to which we need to aggregate the simulation results.
- Phase level. The phase level concerns a single phase, for example, the BS. During this phase, multiple activities have to be carried out, before the process is able to go to the next phase. For a simulation, presumably few data will be available on this level, as these are often not recorded. We therefore suggest using a bottom-up approach and start at the activity level first.
- Activity level. The activity level concerns a single activity, for example, testing during the BS. Although it is not possible to run actual tests on a software product that is not there yet, certain documents will have to be made and agreed on during this phase. For each activity, certain subactivities or steps are performed.

These levels of detail are illustrated in Figure 2. At the project level, six open arrows illustrate the various software production phases. At the next level, each of those six phases includes a number of





activities ranging from testing to configuration management. At the lowest level, each of the activities is broken down into sequential and parallel subactivities, or steps. Each block represents a step during the activity, which starts at the start block, goes on to the subactivities as indicated by the arrows, and finally arrives at the end block, at which time the activity for this phase is completed. For simplicity, only a limited number of steps are shown, which is indicated by the double line before the end block. Some of the steps shown are approvals that will influence the approval duration of the phase in which they are performed.

3.2. Historic benchmark of project durations

In this section, the benchmark values are determined for the KPIs before the outsourcing. In this paper, the project and approval durations of the IT projects are analyzed because the durations alone already illustrate important insights in the new development process after outsourcing. Historic project and approval durations were available in the organization. Although in this paper, we discuss only the project durations, the principles of the analyses apply to other KPIs as well.

In this section, we first examine the characteristics of projects executed with the old development process. To do so, we explored data available within the organization at the project level. For all projects, data on start and end dates for the production phases were available. However, not all dates were entered. Next to that, not all projects were comparable, for example, because some projects did not use DSDM. Therefore, we made a selection of those projects that are usable for our further analyses. We performed additional checks to ensure we have an adequate selection.

Based on the selected projects, we determined the average project duration including the time required for approvals. These values were used as benchmark values to assess the impact of the new development process.

3.2.1. Selecting projects. We explored the total duration of projects in the old situation. From the organization, we obtained access to a database with hundreds of IT projects. The database contained among others information on the start and end dates for each specific phase in the project, project type, project development method, project completion, project manager, developing party, budgets, and function counts.

However, the data were not complete for all projects. Some projects missed certain information or were executed with different development processes. Therefore, a selection of the data was required before being able to assess the overall project duration. We did this by exploring the available data as we will show in this section.

In order to determine the overall project duration, one needs to consider the start and end dates of the various phases that are used in the DSDM process. From Figure 2, one could think that the total duration is calculated by adding the durations of the different phases. However, this is not the case because there turned out to be large overlaps between the different phases. We found overlaps similar to those found in the Rational Unified Process book [14] with similar graphics.

In Figure 3, we visualized the course of a number of the selected projects by plotting the start and end times of each phase for these projects in a parallel coordinates plot ([33], p. 314). The horizontal axis has an ordinal scale representing each phase in an equidistant manner. The vertical axis represents the time. In this figure, a small selection of our projects is plotted, and each line represents a single project, connecting its start and end dates for all phases where data were available. We used a small selection of projects because we wished to illustrate the wide variation concerning each phase.

Basically, the plot provides insight into the large variation and substantial overlaps that reside in each project. We observe a pattern that almost all projects follow, namely that the phases overlap, because the graph shows spikes for all the projects. In this picture, there is also an outlier, shown in light gray that we left out in our final selection. This is a project where the end dates for all phases are equal to the in-production date, which reaches the in-production date line for each phase end that has data. Via the parallel coordinates plot, we discovered these outliers. As these outliers lack the required data to adequately assess the overall project and approval durations, we left them out of the selection.



Figure 3. Start and end dates per phase for some projects.

For our historic data set, we used projects satisfying the following criteria, to make sure we only have comparable projects.

- Only development projects
- · Only projects following the DSDM method
- · Only projects completed successfully
- · Only projects that were developed in-house
- Only projects that had no external budget (as an extra assurance to the aforementioned)
- Only projects that had function point (size) estimates
- No projects where the end date of all phases equaled the end date of the project (representing some outliers)

This selection left us with 318 projects that provided a good characterization of the original development process before the outsourcing. Before computing the historic benchmark of the project durations, we illustrate additional checks we performed to ensure we had an adequate selection.

3.2.2. Checking selection. While Figure 3 provided us early insight into the wide variation and the significant overlap between phases, only a few projects are shown. Figure 4 aggregates this information for all the selected projects. We display the overlaps and durations of the different phases by means of the well-known Gantt chart [11, 25]. Because the projects have different durations, the total duration had to be normalized, in order to aggregate all data on all projects into a single Gantt chart. As time data were not always available for every phase, and especially as data on the FS are very limited (only 10 projects), we normalized the data based on the budget availability date and the in-production date, which were always available. In this way, we normalized all our 318 projects. After normalization, we calculated the median start and end dates for the different phases, which are shown in Figure 4.

On multiple occasions, a project is started before there is a budget. In fact, the median budget availability date is very near the start of the BS, while the in-production date is equal to the end of the IMPL phase. However, after leaving out projects that start before the budget availability date, the Gantt chart did not deviate from the chart displayed in Figure 4. Despite our proxy of dates via budget, this Gantt chart accurately describes the overlap of the different phases in the software production practice before outsourcing.

We also investigated whether projects of different sizes would have produced different Gantt charts by separating between small and large projects, based on function points. The Gantt charts produced for only the small projects or only the larger projects did not differ significantly however. We tested



Figure 4. Median Gantt chart showing overlap and duration of the phases of software production for 318 information technology projects.

this also via a formal statistical test by using an analysis-of-variance test [33], which is shown in Table I. We used the start and end dates of the phases as response variables and the phases and function points as explanatory variables. The table shows that the effect of function points on the response variables has no statistical significance. We took the logarithm for the function points to make the data more normally distributed [35], but this has no influence on the conclusions. For many IT KPIs, the assumption of normality does not hold [34], as is also the case in this analysis.

Because the assumption of normality does not hold for many IT KPIs, the reported *p*-values in the table should be lower in reality, as the kurtosis per group is higher (e.g., [31]). But even if we take this into consideration, it is unlikely that we should reject the hypothesis that the number of function points has an influence on the points used in the Gantt chart. Therefore, we cannot conclude that projects of different sizes should be treated differently.

To produce the normalized Gantt chart, we aggregated the data of many projects, because the dates vary heavily between projects, as we will illustrate in further detail. In Figure 5, we show the start date per phase for all projects in a beanplot [15]. In this plot, the small green lines represent individual measurements, while the overall density is also shown as the shape in lilac. The prominent black lines show the median of the start time per phase.

As is shown in this beanplot, there are some projects that start before there is a budget. With inhouse software development, this is something that is sometimes carried out, as the risks are accountable to the same party anyway. In an outsourcing setting, this will be less common, as an insourcing party then bears the risk of not being paid. Therefore, the process is likely to change after an outsourcing transition, which affects the planning of the projects. This effect can be simulated as shown later in this paper.

Also shown in the beanplot is that the distributions appear to have heavy tails; that is, there are sometimes huge deviations from the mean. The large variation we observed is not an uncommon behavior for IT data as elaborately discussed elsewhere [34, 35]. Recently, data points in the heavy tail are sometimes labeled as 'black swan' events. They are more prevalent than one thinks and cannot be simply rejected from quantitative studies [5]. Because of the large variation in the data, a deterministic approach with fixed times is likely to lead to inappropriate results when simulating such data, and we therefore advise a stochastic approach that does take the large variation into account.

Table 1	l. A	nal	ysis-	of	variance	tab	le	for t	the	relation	between	function	on j	points	and	the	Gantt	chart	point	ts.

	df	Sum square	Mean square	F value	$\Pr(>F)$
Phases Log(function points) Residuals	15 16 2932	266.163 0.368 116.621	17.744 0.023 0.040	446.1125 0.5781	<2e-16 0.9024



Figure 5. Beanplot of start dates per phase relative to budget (=0) and in-production date (=1).

3.2.3. Computing benchmark. Earlier, we described the available data in this organization. We showed how to select comparable projects to compute a reliable benchmark. Based on the selected data, we then calculated the durations of the different phases.

The average duration of the IT projects in the old development process was 164 working days. Although we have information on the duration of the phases, the data did not allow determining of the exact number of days spent on waiting for approvals. Together with experts of the organization, and the data we did have, we estimated this to an average of 20 working days in the old situation. In our simulation of the newly proposed process, for simplicity, we will only take approvals into account because they form the main difference due to changes in process. While it is possible that the other subactivities are performed faster, waiting for approval cannot be improved upon. For the subactivities that were not approvals, no data were available for the old situation.

Using these benchmark values, we can assess the impact of the change in development process on the project duration.

4. SIMULATION

In this section, we determine the impact of the change to the new development process on the project duration. By estimating the project duration with the new development process and the historic benchmark found in the previous section, we will be able to assess the change of process, allowing for a fair assessment of the influence of the outsourcer.

To be able to separately measure the effect of a change in process after a transition, we turn to simulation. Software process simulation [17, 38] enables us to estimate the outcomes of the proposed process, if all the relevant and necessary data to carry out the simulation can be provided by the designers of the new process. Fortunately, this turned out to be the case in our case study.

There are multiple methods to do such a simulation, which we show in Section 4.1. We will illustrate a simple method in Section 4.2. But because the simple method has some drawbacks, in Section 4.3, we will use a more refined method. In Section 4.4, we validate the assumptions on which the simulation is based using real data that became available after an altered but comparable version of the transition was made. In this altered version, considerably fewer approvals were put into the process to reduce negative impacts found by the simulation we discuss in this section. Afterwards, it turned out that our simulation steered decision making into the right direction.

4.1. Simulation of a development process

First, we discuss the different types of simulation. Basically, for simulating a development process, we identify two types of stochastic simulations: plain Monte Carlo simulation and discrete event simulation. We briefly introduce both in the context of simulating development processes.

- Plain Monte Carlo simulation [20]. In this type of simulation, values for times of different events are being drawn from given probability distributions. These values then become subject to simple calculations, like summation and taking the maximum of a set of values. After repeating these calculations many times, conclusions on the behavior of the process are drawn. This type of simulation is best suited for more global simulations, for example, on the project level in a software development process. We will give an example of this type of simulation in Section 4.2. In the literature, we found examples of this type in articles of Raffo [24]. Plain Monte Carlo simulation is supported by a large variety of tools, for instance, by Microsoft Excel and a plug-in like Crystal Ball [9].
- Discrete event simulation. In this type of simulation, scheduled events take place and are executed. Events are, for instance, the start of a project, the approval of a certain step, and the end of a project. The simulation environment allows for the existence of multiple entities at the same time (e.g., multiple software development projects). Next to that, a flowchart visualization of the process at hand is often built in for standard simulation tools.

Usually, building blocks like an entity initiator, a worker with queue, and blocks to combine and separate the work flow are available for building a simulation model. For each step in the process, parameters on the amount of time it takes are settable. Discrete event simulation is used in our case study, mainly because of the ability to visualize the process being simulated via swim-lane diagrams. It also offers extra flexibility; for example, it simulates queuing in case multiple projects have to be handled simultaneously by the same persons. In the case study presented in this article, queuing is not taken into account, however. In conclusion, discrete event simulation is a more detailed form of simulation oriented around the execution of events, which is more suitable for process-oriented applications.

4.2. Drawbacks of plain Monte Carlo simulation

We will illustrate the workings of the plain Monte Carlo simulation at the activity level with Excel by using an example. In Figure 6, we show the use of Excel as a simulation tool for a straightforward process consisting of four steps. First, there are three consecutive steps, followed by one step consisting of three parallel substeps. The average times taken by each step are estimated, and an exponential distribution is assumed. An exponential distribution is in accordance with a situation where the expected waiting time does not change if you already know that you have waited for a certain duration (like throwing a die each day and waiting until you throw a six). A sample from the

0		N -) T	simple sim	ulation example (using excelusisx - Micr	rosoft Excel				_ = ×	c	
	Home	Insert Page Layout Formulas Data Review	View Deve	loper						0 - 5 >	×	
Past	Cut Can Copy	Arial \cdot 10 \cdot A' \cdot $=$ $=$	≫- ■ Wrap 連連 Merge	Text Nu	mber •	Conditional For	mat Cell In	sert Delete Format	Σ AutoSum *	ort & Find &		
*	Clipboard	5 Font 6	Alignment	6	Number 5	Formatting * as Ti Style	sble * Styles *	Cells	Editir	liter * Select *		
D2 • (* // ==LN(RAND())*\$C2										1	¥	
	A	В	С	D	E	F	G	Н	1	J	-	
1		Task	Estimated	Duration 1	Duration 2	Duration 3	Duration 4	Duration 5	Duration 6	Duration 7	ī	
2	1	Send task to FD	0.5	0.45275	0.01922	1.09700	0.53052	0.02947	0.77118	0.11646		
3	2	Get approval back	2.5	5.0419	3.36217	1.96560	6.74996	6.74996 5.66147		1.49643		
4	3	Send task to VM	2	4.77583	0.97690	1.60634	0.37225	0.40619	3.66750	1.06498		
5	4a	Get signature from company	2	0.98096	0.93580	1.07029	1.25848	0.43034	0.05987	4.26497		
6	4b	Get signature from vendor	3	1.52275	4.14683	4.79211	9.56916	1.22072	4.35143	1.98762		
7	4c	Get signature from maintainer	3	3.43739	0.38952	4.48545	0.05423	2.92077	4.65353	8.76479		
8	4 total	Get signatures total		3.43739	4.14683	4.79211	9.56916	2.92077	4.65353	8.7647		
9		Total process duration		13.70794	8.50513	9.46105	17.22189	9.01790	12.56473	11.44266	L	
10		Within 10 days?		FALSE	TRUE	TRUE	FALSE	TRUE	FALSE	FALSE		
11												
12												
13		Average duration	9.95911									
14		St. dev.	4.586053									
15		95% Confidence interval for average	0.089885	->range:	9.869225	10.04899						
16												
17		Service Level (within 10 days)	0.5692									
18		St. dev.	0.495213								L	
19		95% Confidence interval for SL	0.009706	->range:	0.559494	0.578906						
20												
21	► H simula	tion 💫				1 (1					*	
Read												

Figure 6. Screenshot of Monte Carlo simulation with Excel.

exponential distribution is drawn in Excel by using the formula = [average] * - LN(RAND()), with [average] being the mean of the wanted distribution, as shown in the screenshot (Figure 6).

Using the values drawn from the exponential distributions, we obtained an estimate for the complete four-step process. As the fourth step is only completed until all signatures are received, the maximum of three such distributions is used, given by the formula = MAX (D5:D7) (for column D). Totaling the steps (using SUM()), we calculate the total approval duration of the process, amounting to 13.7 working days in this case (given by field D9). This is only a single-point estimate, which of course is not very accurate. Therefore, we have to simulate this process many times. In this example, we do this 10,000 times. For our 10,000 iterations, the average approval time is estimated at 9.96 working days. Of course, we want to know how reliable such an estimate is. An indication is given by a 95% confidence interval. In 95% of such intervals, the sought value is contained. Excel readily provides functionality to calculate such an interval, in this case, by using the formula = CONFIDENCE (0.05, C14, 10,000). Therefore, we have a good indication that the sought value lies between 9.87 and 10.05. If one does more iterations than the 10,000 we did here, this interval will be even smaller.

Suppose 10 days' approval duration is acceptable by management. Then, it seems that this target is met. However, the number of times the target of 10 days is actually met is just 57%, which is not very high.

A solution could be to scrap the signature from the company and the signature from the maintainer, leaving only one necessary signature at step 4. Doing so leads to an estimated service level of 73%, which is much more acceptable. Using Excel, the answer to such a what-if scenario is produced with ease (by filling in a zero in cells C5 and C7).

This simple method of simulation also has some disadvantages. For example, it is quite hard to put rejections and other 'loops' into the spreadsheet model. Also, it is hard to run multiple processes together into the model to do stress tests and obtain an idea of the queuing that occurs, for example, if one wants to observe the effect if everybody needs approval from the same person at the same time. Next to that, visualization is not offered by Excel, and some advanced statistical analysis is not easily available, at least not without plug-ins. Because of these disadvantages, we advocate the discrete event simulation. We used this technology for our final simulation.

4.3. Simulating our case study

In Section 3.2.1, we have shown that there is presently an overlap between the DSDM phases. Because of this overlap, the duration of a project cannot be calculated by adding the durations of the different phases. This also means that when we simulate each phase separately, it is not immediately clear how to aggregate the total duration from its parts. However, we found out from the organization that in our simulation, this is not a problem. In our case study, the proposed process is more formal and involves many parties, which implies that, say, the testing company will not start before an official approval, if only to avoid invoicing problems. Moreover, before a next phase is commenced, usually approvals of more than two parties are necessary, because in our case, a multiparty setting is at hand. Therefore, we aggregated the total duration of the approvals in the new situation by simply adding the durations of the parts. Note that we only took approvals into account in this aggregation, but it is likely that some of the other activities also overlapped in the old process, making the results of our simulation conservative.

In our case study, a new development process is proposed, which by definition could not have been utilized. The organization has designed this proposed process, and one of the design artifacts is a so-called swim-lane diagram [14]. Within this diagram, various moments for approvals are modeled. The designers of the diagram were able to estimate the duration of individual approvals.

For our simulation, we used a simulation environment called Arena [18], a commercial system from Rockwell Automation, Inc. In this program, we encoded the swim-lane diagrams that were designed for the outsourced situation. In Figure 7, we depict the result.

Each of the squares represents a step in the activity like the steps in the activity level in Figure 2. Figure 7 shows a simulation at the activity level focused on the activities during the DBI, with different actors shown in a swim-lane diagram. We also carried out such simulations for the BS and the FMI.



Figure 7. Screenshot of the simulation environment that we used to simulate the proposed software development process after outsourcing.

As previously mentioned, we only simulated the approval durations in our case study because of simplicity and availability of data. So, we assumed each step of the process to have zero duration, except for the approvals, as these make the difference between the old and newly proposed processes. In total, tens of approvals were modeled.

The organization identified two types of approvals, one to sign off documents or decisions and one for reviews. The sign-offs were expected to take two to four working days and the reviews three to five working days. We modeled both these approval durations with triangular distributions, because the mean values of three and four working days were expected to occur most frequently. Also, the probability of hitting the outer values was originally expected to be very low, while exceeding the outer values should not occur, which is also the case for a triangular distribution.

Rejections, rework, and re-approvals were not taken into account in these estimated approval durations. The organization expected that these effects would not influence the process much; later on, we will see that they were right about that but that, in some cases, the approval durations were much higher than expected. In all three cases, we checked with the designers of the newly proposed process whether the simulation model represented the different actors shown in the swim-lane diagram correctly, and they agreed.

4.4. Validation of the simulation

An important aspect of any simulation is the validation of the results. Before we show the results of our simulation, we therefore first validate the major assumptions on which the simulation is based. Because the process that was simulated in this paper was never put into practice, and we cannot show the actually implemented process because of confidentiality, we cannot directly validate the final outcomes of the simulation. It is, however, possible to test the important assumptions on which the simulation was based. Namely, after IMPL, we gathered data that were automatically collected by the work flow management software used in the new process, an altered but comparable version of the proposed process we simulated. We base ourselves on 1 year of data on 761 development projects, which had multiple approvals per project and followed the same approval process.

In this section, we compare the real-world data with our simulation. The data have a working-day accuracy, because working-hour accuracy was unavailable and is also hard to define because of differences in working times, for example, because work is performed in different time zones, which makes it unclear which hours should be considered working hours.

We validate three major assumptions made, which all turned out to be valid for conservative scenarios. The first major assumption that will be tested is the duration of the approval times. It was assumed that a review would take three to five business days. In Figure 8, the solid line shows the distribution of the approval duration for a review step in the process that was finally implemented, an improved version of the proposed process that we simulate. The median of the number of business days measured is four business days, which is in agreement with the three to five business days that was assumed. There were some reviews that take less than 3 days, so the distribution is in fact not really triangular between three and five business days. A large part of the distribution is reasonably modeled by a triangular distribution between 0 and 8 days, as shown by the dashed line, so a triangular distribution with a mean of four business days models common situations. The high end of the distribution is, being over 60 days, larger than was expected. This means that the results of our simulation will be conservative. In the real-world data displayed in Figure 8, rejections, rework, and re-approvals are present. In our simulations, we did not incorporate them because they were estimated not to influence the 3-5 days much. This assumption is therefore true. Of course, leaving out this assumption, the location of four business days for the expected approval duration holds, while the variance of one business day in a triangular distribution is only true for conservative scenarios.

The second major assumption is that the approval durations of different steps in the process are independent of each other. However, for two steps that were actually put into practice, a formal statistical test showed that they were dependent. We initially obtained a p-value of 0.0016 for Kendall's rank test [33, 7], which indicates that there was a correlation. The reason for this was that we originally put our full data set of 1762 projects into one group, but some projects follow a special process, in which these steps were either more complicated (for the largest projects) or simplified (for small change requests). After removal of small, amendment, and special (very large) projects, we were left with 761 normal projects, and there was no longer an indication for correlation between steps. The smallest p-value we obtained was 0.12, indicating no correlation. We also did not spot any correlation after plotting the data. In Figure 9, the approval durations for two of the largest steps in the implemented process are shown, indicating no correlation. For the majority of the projects, the second major assumption of the simulation holds. It does not hold for the complete portfolio because in the process that was finally implemented, some refinements were made for special projects. In the proposed process that we simulate, there are no special projects. Therefore, the assumption of independent approval durations in the simulation holds.



approval duration for a review step (days)

Figure 8. Density plot of one of the actual approval durations in the implemented process. Some random uniform noise (jitter, [6]) is used to show overlapping points. The dashed line is a triangular distribution at the same location as in our simulation, but its variance is higher.

no correlation between approval durations



Figure 9. Scatter plot of one of the actual approval durations for two reviews in the newly implemented process. Some random uniform noise and a log–log scale are used to show overlapping points.

A third assumption is that there are no upward or downward trends in the data, because we will simulate a stable situation and give stable results.

If there is a downward trend in the data, for example, because the process becomes better understood and the time it takes to complete a step decreases, our simulation should not return stable results. In Figure 10, we show approval durations over time for a single step right after implementation of the new process. We show the average duration as the dotted line and a normal kernel smoother [33] as the solid trend line. It is clearly visible that there are correlations over time; the trend line is not very flat. Approval times of subsequent projects tend to be more equal than if you would take two random projects. Queuing is a likely cause of such correlations, which because of simplicity, we did not implement because it will almost average out. Over a longer period, queuing in a stable queue only increases the time spent waiting for approval, which we *did* take into account in our predictions for the waiting time. The trend line shows no upward or downward trend and just fluctuates around the mean; otherwise, we should have adapted the simulation to take trends into account. Using a formal statistical test for trends, namely trend.test [13], with a p-value of 0.759 for a Spearman rank correlation test and a p-value of 0.356 after 1000 runs of its bootstrapped variant, indeed no statistical evidence was found for increasing or decreasing trends. Therefore, the approval duration is stable enough to not severely affect the outcomes of a simulation such as the one carried out in this paper.

A final assumption, which unfortunately could not be directly validated, is that for the purpose of comparison of processes, we only need to take into account the differences in approval duration. There are no data available to directly test this assumption. Based on a qualitative analysis of the



Figure 10. Actual approval durations for a single review over time show a correlation but no upward or downward trend.

activities and their positions within the processes, we found this assumption reasonable for our simulation, as the main difference between the two processes was formed by time taken for approvals, and a better approach was not available because of a lack of data.

5. RESULTS

Using the data and methodology of the previous sections, we simulated the approval durations of the BS phase, the FMI phase, and the design-and-build phase, because for these three phases, a new process was proposed. Using the *ex post* validated assumptions described in Section 4, each of the phases was simulated 10,000 times. By doing this 10,000 times, we obtained reliable estimates. Figure 11 depicts the results of the simulation. The proposed process spends on average 20 waiting days for approvals for the BS, another 21 for the FMI phase, and 26 days for the design-and-build phase.

The average waiting time for approvals is between 26 and 67 days depending on the overlap. We learned from the organization that the individual approval durations had to be added because of the formalities in the proposed process, which resulted in a total approval duration of 67 days.

In the original situation, we had an expert estimation of on average 20 days spent in total on approvals. Note that the old process contained a single formal approval point providing for a final go/kill decision. The formalization of the process thus introduces on average 67 - 20 = 47 days' extra approval duration. So, a project takes on average 47 working days more because of extra approvals. Given that the mean duration of a project using the old development process is 164 days, this means an increase of 47/164 = 29% in duration. So the correction factor as stated in Section 2 for time to market due to changes in process needs to be 1.29 in case the proposed new development process will be implemented.

We consulted the organization about this large increase in approval duration, and it was clear that this was an unwanted situation corroborating their intuition that the process contained too much regulatory overhead. The time to market would simply not be acceptable. Based on our simulation and their intuition, the organization chose to use a less formal process in case of smaller projects and adapted the proposed process to decrease approval times.

Our analysis illustrates that with simulations, we were able to estimate the possible effects after transition at the time of contracting, before a single project was carried out using the proposed process. For the proposed process, we were able to estimate the impact of process changes. This enabled the effect to be accounted for in the negotiations on the quantitative measures in the contracts. This in turn helped in conflict prevention, which is in the interest of both the offering organization and the other parties involved in the outsourcing.

It took three person days to obtain the simulation result, and the statistical analysis took about six to seven person days. So with a marginal amount of effort, sufficient quantitative insight was created to steer the outsourcing transition, to prevent conflicts, and to support decision making about pragmatic modifications to the new development process.



Figure 11. Results of the simulation.

6. SIMULATION OF OTHER METRICS

In our case study, we analyzed time to market. In the introduction, we mentioned that also other metrics, in domains such as size, cost, quality, and productivity, are affected as a consequence of changing the software process. In this section, we will briefly touch upon these other metrics.

Productivity For the purpose of this paper, we think of productivity in terms of size divided by effort. We prefer to use a reliable size measure such as function points (e.g., [16]), in this case, function points per thousand hours.

Any extra effort spent because of a change in the development process directly translates into a lower productivity. If the amount of staff remains stable during the project, then an increase in project time also directly translates to a decrease in productivity, even if the staff is just waiting for approval. In this scenario, the same correction factor of 1.29 as we found for time to market is applicable. The same way of reasoning is also appropriate if one considers cost per function point. The cost metric has, however, some extra difficulties in analysis, especially if the cost effect of software productivity tools is taken into account. For instance, the new development process can use a less expensive tool that will increase the effort but does not increase the cost, thereby not changing productivity in terms of function points per monetary unit.

Theoretically, staff could do other work perhaps on a different project so that no valuable time is lost. However, this will only resolve lost productivity due to approval durations to a very limited level. For it takes some time for knowledge workers to get into the so-called flow – a highly focused state of consciousness in which things go as if automatic and effortless ([8], pp. 110–113). Disturbing this flow, or partitioning knowledge work over too many different tasks, leads to no work effectively being carried out at all, also when it comes to IT work [10]. Therefore, large approval durations lead to time decompression [36], of which we know already for decades that it will negatively influence effort put in a project; it takes more effort to make the same functionality if you take too much time for it ([3], p. 472).

If the staff does not remain stable and less people are involved at a certain moment, the change in productivity is less dramatic. This would make the shock in productivity less prominent. Our model is easily extendable by incorporating the time at which an approval occurs and adjust for this time, for example, by using a Rayleigh distribution (e.g., [3]).

The simulation also shows that the proposed process had signs of overregulation. In a paper by Verhoef [36], overregulation is found to have a significant influence on the productivity. In that paper, an organization that had a low productivity compared with the benchmark found among peers is described. That case study also found that 20% of the total project duration consisted of waiting for approvals, which turned out to explain a significant amount of the difference in productivity. The simulation in our case study shows that in the proposed process, an estimated 67/(164+47)=32% of the total project duration will consist of waiting for approvals.

Size Size is usually unaffected by a vendor because the client is normally the one who decides on the scope of projects. Changes in size should therefore be attributed to a change in process. It could, for example, be that smaller projects are launched because projects of a certain size are allowed to follow a different process. It could also be that projects are consolidated into one giant project so that the regulatory part is only necessary once. An effect that could be caused by a vendor is that larger projects are implemented because the budget now allows it, but in our experience, this effect is limited.

Quality Quality is affected by both the change in process and the change in supplier. Quality improvements or deteriorations due to the change in process can be estimated by impact studies of similar measures. In theory, simulations could be used to measure the impact of the introduction of various (approval) measures into a process, but in practice, it might be hard to gather hard data on the quality that such measures add.

Cost Cost is affected by both the change in process and the change in supplier. The effects caused by the change in process can be attributed by breaking down the changes in different steps. The costs of waiting and doing nothing have to be counted as well, for which a simulation of the increase in waiting time is useful.

7. USES FOR SIMULATION

So far, we have mainly illustrated how simulating a proposed process helps in finding a correction factor for discontinuities in IT metrics due to changing the process. But simulations of new proposed processes are also useful in resolving other issues and gaining insight into the workings of a proposed process before it is implemented. Some situations in which the use of simulation is useful are outlined in the following.

- Simulation provides answers to questions such as whether the time needed for certain approvals is acceptable. Can the time needed be reduced by removing approvals or by adjusting the point in time where the approval is performed in the process? How much time can be saved if not five but only three persons approve a document at a certain step? We saw an example of such an analysis where we removed two signatures in order to improve the SLA level.
- Simulation can also be used to find bottlenecks in the newly proposed process. What if a certain step in a phase can only be carried out by one person? What will happen if 10 projects that all need that same person's approval are scheduled close to each other? How will this affect the total duration of these projects? Could this bottleneck be solved by adding another person for this task or are more needed?

8. DISCUSSION

An advantage of simulation is that it is possible to model any real-life aspect of a process into the simulation. This is also a disadvantage because models tend to become very complicated, hard to implement, and incomprehensible. Or, as Box stated in 1976 [4]: 'Since all models are wrong the scientist cannot obtain a "correct" one by excessive elaboration. On the contrary following William of Occam he should seek an economical description of natural phenomena. Just as the ability to devise simple but evocative models is the signature of the great scientist so overelaboration and overparameterization is often the mark of mediocrity.'

To prevent this overelaboration, models should be kept simple. In this case study, we also made numerous simplifications. For example, we did not take the time into account to create the extra documents needed for the process that was proposed. Extra documentation increases the total time to execute a project in the new situation. Another simplification is that we only focused on approvals and did not include other activities that take place at the same time as well. We did not implement queuing, which is a cause of extra delays. Therefore, because of the simplifications, our estimates are most probably conservative.

Despite the simplifications, we believe that having such estimations is far better than having no estimation at all, or having an overelaborated model. In fact, it can be argued that a simpler simulation is easier to understand and more time efficient. In our case, a simulation of a proposed development process led to a change of plan, and a simpler process was implemented.

Finally, we must warn that a negative change in an IT metric such as productivity in terms of function points per dollar is not necessarily a bad situation. For example, if the amount of function points produced per day goes down, it could well be because the approvals in the improved process forced the organization to put more emphasis on requirements engineering and risk management. This way, it is more likely that instead of building unnecessary functionality, only the essential function points are put in production, instead of mindlessly producing functionality. Also, more approvals allow for more moments to intervene in a project and assess whether the risk of failure for

the project is still within bounds. If the expected risk is too high, risks should be mitigated first before proceeding further. Another important aspect is the quality of the software produced, which is often a reason for introducing a more formal process. Although this may lead to lower productivity, it is certainly a better overall situation. Similar arguments apply to the other metrics.

9. CONCLUSIONS

Often, an outsourcing contract contains quantitative targets for productivity, cost, quality, and time to market. To prevent future disputes about meeting these targets, their effect on the transition needs to be taken into account. We have shown that an outsourcing transition affects IT metrics in two ways. First, they are influenced by changes made in the development process. Second, they are affected by the change of supplier. For assessing the change in IT metrics due to the change in the development process, we proposed a simulation method that calculates a correction factor to assess the performance of the supplier in a fair way.

In this paper, we illustrated a real-world case in which simulation was used to quantify changes in IT metrics after outsourcing transitions. Actual data supported the assumptions on which the simulation was built. Using the simulation, we were able to advise the organization successfully that their proposed development process had to be adapted so that the balance between a more formal process and time to market was more in line with their demands.

ACKNOWLEDGMENTS

This research received partial support from the Dutch Joint Academic and Commercial Quality Research & Development (Jacquard) program on Software Engineering Research via contract 638.004.405 *Equity: Exploring Quantifiable Information Technology Yields* and contract 638.003.611 *Symbiosis: Synergy of managing business-IT-alignment, IT-sourcing and offshoring success in society.* We are also grateful to the organizations that provided us with their data.

REFERENCES

- Ali N, Petersen K. A consolidated process for software process simulation: state of the art and industry experience. In 38th EUROMICRO Conference on Software Engineering and Advanced Applications (SEAA), 2012; pp. 327–336.
- Birkholzer T. Software process simulation is simulation too what can be learned from other domains of simulation? In 2012 International Conference on Software and System Process (ICSSP), 2012; pp. 223–225.
- 3. Boehm B. Software Engineering Economics. Prentice Hall: Englewood Cliffs, NJ, 1981.
- 4. Box GE. Science and statistics. Journal of the American Statistical Association 1976; 71(356):791–799.
- 5. Budzier A, Flyvbjerg B. Making-sense of the impact and importance of outliers in project management through the use of power laws. Proceedings of IRNOP (International Research Network on Organizing by Projects), at Oslo, 2013; 11.
- 6. Chambers J, Cleveland W, Kleiner B, Tukey P. Graphical Methods for Data Analysis. Wadsworth, 1983.
- Conover W. Practical nonparametric statistics. Probability and Mathematical Statistics (3rd edn). John Wiley & sons: New York, 1980.
- 8. Csikszentmihalyi M. Creativity Flow and the Psychology of Discovery and Invention. Harper Perennial, 1996.
- 9. Decisioneering, Inc., Denver, CO. Crystal ball, computer program and users guide, 1993.
- 10. DeMarco T, Lister T. Peopleware Productive Projects and Teams. Dorset House, 1987.
- 11. Gantt H. Organizing for Work. Harcourt, Brace & Howe: New York, 1919.
- Gopal A, Mukhopadhyay T, Krishnan MS. The role of software processes and communication in offshore software development. *Communications of the ACM* 2002; 45(4):193–200.
- Ibanez F, Grosjean P, Etienne M. Pastecs: package for analysis of space-time ecological series, 2009. R package version 1.3-8.
- Jacobson I, Booch G, Rumbaugh J. The Unified Software Development Process. Object Technology Series. Addison-Wesley, 1998.
- Kampstra P. Beanplot: a boxplot alternative for visual comparison of distributions. *Journal of Statistical Software, Code Snippets* 2008; 28(1):1–9, 10.
- Kampstra P, Verhoef C. Reliability of function point counts, 2009. (Available from: http://www.cs.vu.nl/~x/rofpc/ rofpc.pdf)
- 17. Kellner M, Madachy R, Raffo D. Software process simulation modeling: why? what? how? *Journal of Systems and Software* 1999; **46**(2):91–105.
- Kelton D, Sadowski R, Sadowski D. Simulation with Arena. McGraw-Hill Series in Industrial Engineering and Management Science. McGraw-Hill, Inc.: New York, 2003.

- 19. Malhotra Y. Business process redesign: an overview. IEEE Engineering Management Review 1998; 3(26):27–31.
- 20. Metropolis N, Ulam S. The Monte Carlo method. *Journal of the American Statistical Association* 1949; **44**(247):335–341.
- Moe NB, Šmite D. Understanding a lack of trust in Global Software Teams: a multiple-case study. Software Process: Improvement and Practice 2008; 13(3):217–231.
- 22. Plunkett Research. Outsourcing & offshoring industry market research. Website, 2014. (Available from: http://www. plunkettresearch.com/outsourcing-offshoring-bpo-market-research/industry-and-business-data)
- R Development Core Team. R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing: Vienna, Austria, 2008. ISBN 3-900051-07-0.
- 24. Raffo D, Vandeville J, Martin R. Software process simulation to achieve higher CMM levels. *Journal of Systems and Software* 1999; **46**(2):163–172.
- Rathe A (ed.). Gantt on Management: Guidelines for Today's Executive. American Management Association: New York, 1961.
- 26. Rivera J, van der Meulen R. Gartner says worldwide IT outsourcing market to reach \$288 billion in 2013. Website, 2013. (Available from: http://www.gartner.com/newsroom/id/2550615.)
- Sims O. Business Objects: Delivering Cooperative Objects for Client–Server. McGraw-Hill, Inc.: New York, NY, USA, 1994.
- Stapleton J. DSDM: Dynamic Systems Development Method: The Method in Practice. Addison-Wesley Longman Publishing Co.: Inc., Boston, MA, USA, 1997.
- 29. Stapleton J. DSDM Business Focused Development. Addison-Wesley 2nd edn, 2003.
- 30. Stapleton J, Constable P. DSDM Dynamic System Development Method. Addison-Wesley, 1997.
- 31. StatSoft, Inc. Electronic statistics textbook, 2003. (Available from: http://www.statsoft.com/textbook/stathome.html.)
- Sutton S. Advancing process modeling, simulation, and analytics in practice. In Software and System Process (ICSSP), 2012 International Conference on, pp. 221–222, 2012.
- 33. Venables W, Ripley B. Modern Applied Statistics with S. Springer Verlag 4th edn, 2002.
- Verhoef C. Quantitative IT portfolio management. Science of computer programming, 45(1):1–96, 2002. (Available from: http://www.cs.vu.nl/~x/ipm/ipm.pdf.)
- 35. Verhoef C. Quantifying software process improvement, 2004. (Available from: http://www.cs.vu.nl/~x/spi/spi.pdf.)
- 36. Verhoef C. Quantifying the effects of IT-governance rules. Science of Computer Programming, 67(2–3):247–277, 2007. (Available from: http://www.cs.vu.nl/~x/gov/gov.pdf.)
- 37. Zhang H, Kitchenham B, Pfahl D. Reflections on 10 years of software process simulation modeling: a systematic review. In Making Globally Distributed Software Development a Success Story, Volume 5007 of Lecture Notes in Computer Science Wang Q, Pfahl D, Raffo D (eds.). Springer: Heidelberg, 2008; 345–356.
- Zhang H, Kitchenham B, Pfahl D. Software process simulation modeling: facts, trends and directions. Asia-Pacific Software Engineering Conference, 2008; 59–66.