Empirical Analysis of the Performance-related Risk

Andreas Schmietendorf*#, Reiner Dumke*

* Otto-von-Guericke-Universität Magdeburg, Fakultät Informatik, Institut für verteilte Systeme, Postfach 41 20, D-39016 Magdeburg, E-Mail: {schmiete, dumke}@ivs.cs.uni-magdeburg.de

\# T-Nova Deutsche Telekom Innovationsgesellschaft mbH, Entwicklungszentrum Berlin, Wittestraße 30N, D-13476 Berlin, E-Mail: A.Schmietendorf@telekom.de

Abstract

The current methods of effort estimation frequently take indirect account of the tasks of software performance engineering (SPE), and provide widely differing conclusions. In order to create transparency and acceptance for this task that has been growing in importance for years in the context of the life cycle of IT systems, an approach (PRM Performance Risk Model) is adopted that derives resource requirements from a corresponding risk analysis, and conversely looks at the business system to be supported, the software development and the operational environment. After a short introduction of the current situation and a look behind the PRM-model itself, this paper describes first experience by the use of the PRM-model within 6 industrial projects.

1 Introduction

To develop and introduce information systems which meet both functional and qualitative requirements, it is necessary to plan appropriate technical and human resources to implement the tasks involved. In general, it is easy to understand that a system with high quality requirements involves greater costs than one with lower quality requirements. However, if one examines the quality feature of space- and time-related efficiency (performance in general) as defined by the ISO 9126 quality standard, it then becomes more difficult to make an assessment. It is necessary to apply software performance engineering (SPE) methods during the entire software development process to guarantee this quality feature.

According to the thesis of [Smith 1990], SPE tasks should be an integral component of software engineering. At the first "Workshop on Software and Performance" [WOSP 1998], the thesis goes a step further and declares that software engineering without performance engineering is not software engineering. This thesis accordingly suggests subjecting the approaches, models and methods of software engineering currently in use to more detailed analysis in terms of their representation of SPE-relevant tasks.

The following figure shows typical times when performance analysis methods are applied.

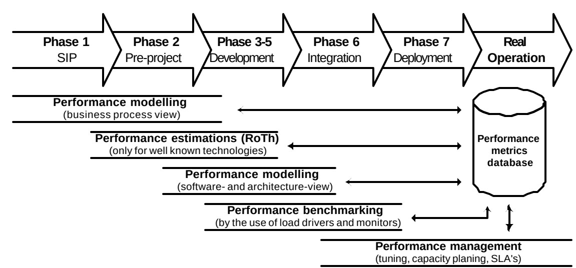

Figure 1: PE-task within the life cycle

When developing software systems with more or less restrictive performance requirements, the standard approach currently adopted usually consists in performance features being examined towards the end of the development phase in the course of acceptance tests, or during pilot operation in the course of multi-stage introduction. The identification of performance shortcomings at the end of software development thus frequently leads to complex tuning measures, an expensive redesigning of the application, the utilisation of hardware with a higher capacity than originally planned or, in an extreme case, to the abandonment of the productive use of the entire information system. Apart from this expenditure which, in general, is not planned, costs especially arise due to the delay in the introduction of the information system and because of problems in the performance of actual functions during active operation, as the business process covered by the IT system is not being supported as planned. There are various technical, methodical and economic reasons for this approach. One problem refers to the inadequate transparency of costs which makes it difficult for project management to understand which SPE tasks are required.

2 Related works

On the last IWSM2000 [Foltin/Schmietendorf 2001] we presented an analysis about well known effort estimations methods and there relations to the efforts of software performance engineering tasks. It was shown that the current methods of estimating costs frequently only indirectly take software performance engineering (SPE) tasks into account. Moreover, there are major differences in the propositions put forward by such methods. The following table shows in conclusion how far different methods of effort estimations take the expenditure for SPE into consideration.

Table 1: effort estimation and SPE

	method	of a project budget for SPE
Expenditure estimate methods	Function Point	max. 10%
	TPA® (Test Point Analysis)	max. 10% of the test budget (black box)
	Object Point	max. 33%
	Cocomo	max. 150%
Expert reports	Connie Smith	usually below 1%
	Capers Jones	average 3%

Apart from the disadvantage of all the different procedures inaccurately and inconsistently mapping the costs involved in software performance engineering, they also do not enable a direct identification to be made, which is the main reason why project management tends to disregard SPE tasks. A description about the mentioned methods ca be found in [IFPUG 1994], [Pol et.al. 2000], [Sneed 1995], [Boehm 1997], [Smith 1990], [Jones 1997].

3 Assessment model

To improve the current situation, a model is proposed to classify performance-related risks [Schmietendorf et al. 2001]. The so called *Performance Risk Model* (PRM) considers three areas involved (business process, development, operational environment), where the occurrence of a performance risk leads to potential losses. These include primary risks R_{pG} arising in connection with the business process, and secondary risks Rs arise in the context of development R_{sE} and the operational environment RsW. The evaluation model presented below is for determining the potential risks in the areas affected.

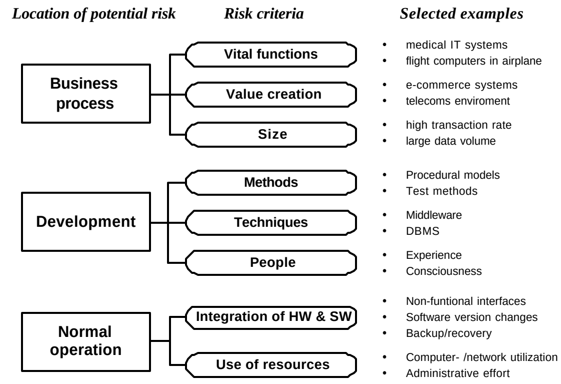

Figure 2: Performance Risk Model (PRM)

We pursue two approaches to quantify the risks in a monetary context. One for the primary risks R_{pGi} (business process) and another one for the secondary risks R_{sEi} (development) and R_{sWi} (real operation). The total risk R (in *Euro*) is the summation of determined primary and second risks weighted via the entry probability p_i.

$$R[Euro] = \sum_{i=1}^{n} p_i R_{pG_i} + \left\{ \sum_{i=1}^{n} p_i R_{sE_i} + \sum_{i=1}^{n} p_i R_{sW_i} \right\}$$

The overall risk R (in Euro) is made up of the summation of primary and secondary risks weighted using the probability of occurrence p_i.

The valuation model should be applied several times in the course of the life cycle of an IT solution, in the form of checklists to be filled in for each risk criterion. This enables risks involved in the business process to be recognised during an SIB (Strategic Information Planning, cf. Business Process Reengineering) for the first time. However, risks which refer to SW development and subsequent active operations are not recognised until the beginning or during the actual development project.

4 Empirical analysis

4.1 Tailoring and Preparation

Within the project PerfEng [PerfEng 2000] we used the new PRM valuation model for empirical studies within 6 industrial projects in the field of telecommunication. In accordance with the procedure implied with the PRM, the following steps were carried out:

1. Work out of a project-specific check list (similarly a tailoring activity).
   - Determination of potential risk categories in dependence on affected fields.

- Specialisation of the risk categories through the actual risk criteria.

2. Carrying out of interviews with representatives of the customer-, developer- and operator-side to the identification of the corresponding performance risk metrics.
   - Identification of organisational information (partic ipants, date, project-phase,...)
   - Information for all participants about the used PRM valuation model.
   - Contents -related introduction of the project to be analysed.
   - Common analysis of the validity of potential risk criteria.
   - Summarising remarks about performance-related project experiences.
3. Statistical analysis of the registered data for the identification of primary risk problems.
   - Identification of cluster frequencies via the evaluated projects
   - Derivation of the corresponding monetary performance risks
   - Determination of the corresponding SPE activities for minimization of performance-related risks.
4. Verification of the valuation model as well as identification of potential improvement potentials both of the valuation model and of the employed check list procedure.

4.2 Achieved results

The following diagram shows in summary results of a use of the proposed PRM valuation model within the projects P1 – P6.

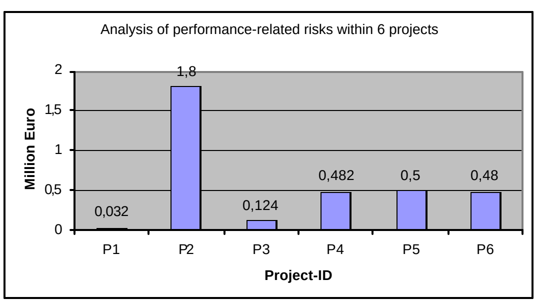

Figure 3: Results of the PRM application

Another evaluation shows the following diagram. The x-axis stands for the expenditures in person months, the y-axis for the monetary performance risks determined in the analysed projects. It shows, that no relation between the size of a project and the effort for SPE tasks exists.

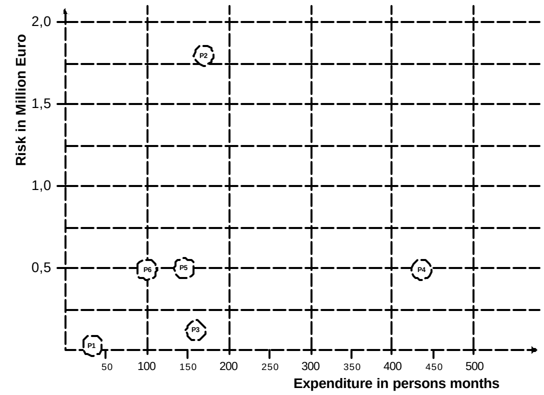

Figure 4: Risks and project size

The most frequent causes of risks (monetary evaluation) referring to the performance of the system to be developed were in:

1. No detailed description of the business process
2. Missing knowledge of new technologies
3. No requirements for the performance behaviour

4.3 Detailed analysis

At the following, all the objects of measurement are shown in detail. With her aid it was possible to identify the corresponding monetary risks within the realised interviews.

M_1 The business process supported by the implemented IT system was not modelled.

M_2 The level of detail of the modelled business process is too small.

M_3 The environment of the business process were not analysed.

M_4 The requirements from the customer-side contains no statements about the amounts of data.

M_5 The requirements from the customer-side contains no statements about the necessary performance behaviour.

M_6 Missing performance requirements were the reasons for change requests.

M_7 Potential backup/recoveries activities lead to performance risks.

M_8 Overlapping of batch- and online-times lead to performance risks.

M_9 Missing synchronisation of parallel user accesses leads to performance risks.

M_{10} The capacity of the hardware-systems within the real operation is to low.

M_{11} Missing knowledge with regard to the performance behaviour of new technologies.

M_{12} Internet based user interfaces implies a performance related risk.

M_{13} The required bandwidths of the used networks implies a performance risk.

M_{14} The used compiler or interpreters lead to performance risks.

M_{15} The used security mechanisms implies performance risks.

The following diagram shows how often the previously mentioned measurement object were identified within the 6 analysed projects.

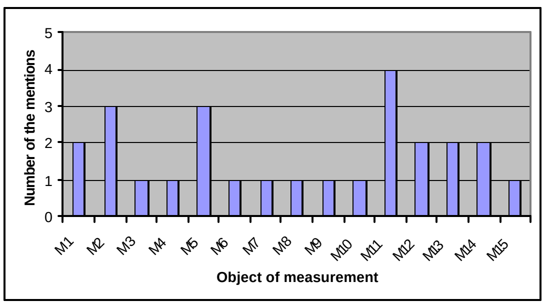

Figure 5: Distribution of the reasons for risk

4.4 Identified Improvement-potentials

During the interviews, it turned out to be very difficult to identify the business case of the applications. The information most frequently given related to the potential savings in personnel and less to statements which referred to the business process to be supported, e.g. the products sold per time unit. Moreover, the information was often gained from experience with subsequent efforts to improve the performance of an information system already in existence. For example, an extension of the project term by using new network and database technologies was a typical case.

A few potential improvements to the proposed valuation model are briefly listed below:

1. The analysis presented in this paper considered mainly projects with object-oriented development technologies. An attempt should be made to broaden the background of empirical experience and, if necessary, effect an appropriate clustering (e.g. SAP, Cobol projects, ...).
2. The contents of the checklists for the interviews should be left open so that suggestions can be included at any time. This procedure was already adopted in the interviews we carried out.
3. To collect performance risk metrics over the course of several projects and evaluate these on a statistical basis, a suitable basis of information must be developed which should best also include other performance-relevant metrics.
4. The valuation model should be applied several times within the software life cycle in order to successively gain more accurate results. Applying it two or three times appears to be worthwhile at the outset.

5 Conclusion

The idea proposed in this article of deriving the costs required to realise performance engineering tasks from potential risks has the advantage of enabling risks to be made transparent and of putting these into monetary terms as far as possible. On the basis of assessed risks it should be easier for project management to integrate SPE tasks into the time schedules and costs of project plans and estimate their added value. Moreover, the selection of actual SPE methods is supported by the definition of a costs framework.

It goes without saying that attempts should be made to generally determine qualitative risks in addition to performance-relevant risks and to integrate this methodology into standard cost evaluation procedures such as the function point procedure. However, it should continue to be possible to explicitly allocate the required costs to performance engineering so that the additional tasks of SPE are not considered as additional costs under any circumstances.

Efficiently operating service providers who, on the one hand, are familiar with performance engineering methods and, on the other, also with those of software engineering are required to carry out performance engineering tasks. With regard to the costs, it has proved wise to make complex SPE tasks, which require a high level of skill and extensive hardware and software equipment, available at a suitable centralised Competence Center. Smaller development companies should resort to the services offered by external providers.

6 References

[Boehm 1997] Boehm, B. W.: COCOMO II Model Definition Manual. University of Southern California, 1997

[Dumke 1998] R. Dumke, E. Foltin, A. Schmietendorf; Kausalitätsprobleme bei der Aufwandsschätzung in der Softwareentwicklung und –wartung, Preprint Nr. 13, Otto-von-Guericke-Universität Magdeburg, August 1998

[Foltin/Schmietendorf 2001] Foltin, E.; Schmietendorf, A.: Estimating the cost of carrying out tasks relating to performance engineering, in R. Dumke et. al., New Approaches in Software Measurement, LNCS 2006 S.111-124, Springer 2001

[IFPUG 1994] IFPUG: Function Point Counting Practices Manual, Atlanta, 1994

[Jones 1997] C. Jones; Software Quality – Analysis and Guidelines for success, International Thomson Computer Press, 1997

[PerfEng 2000] Schmietendorf, A. et. al.: Project PerfEng: Parts 1-7, Internal Report, T-Nova, Development Centre Berlin, 2000

[Pol et.al. 2000] Pol, M.; Koomen, T.; Spillner, A.: Management und Optimierung des Testprozesses, dpunkt.verlag, Heidelberg, 2000

[Schmietendorf et al 2001] Schmietendorf, A.; Foltin, E.; Dumke, R.: Risk-driven effort estimation of tasks within the Software Performance Engineering, In Maxwell, K; Oligny, S.; Kusters, R.; Veenendaal, E. (Eds.): Project Control: Satisfying the Customer, Proc. 12th European Software Control and Metrics conference p.87-97, Shaker Publ., Maastricht/NL 2001

[Schmietendorf/Scholz 1999] Schmietendorf, A.; Scholz, A.: A risk-driven Performance Engineering Process Approach and its Evaluation with a Performance Engineering Maturity Model, in Proc. of the 15th UKPEW, Bristol, 1999

[Smith 1990] C. U. Smith; Performance Engineering of Software Systems – The SEI series in software engineering, Addison-Wesley, 1990

[Sneed 1995] H. M. Sneed; Schätzung der Entwicklungskosten von objektorientierter Software, SES Software-Engineering Services GmbH, Ottobrunn, München 1995

Raw data of the PRM analysis

	Quantified effective risk R_{Si} in Euro (projects P1 until P6)						Cluster frequency (all projects)	R_{Si} in Euro ever object of measurement M_i
Objects of measurement M_i	**P1**	**P2**	**P3**	**P4**	**P5**	**P6**		
Phase	**E**	**W**	**W**	**P**	**W**	**V**		
M_1	2.500	20.000					2	22.500
M_2	2.500	1.200.000		125.000			3	1.327.500
M_3				200.000			1	200.000
M_4				7.700			1	7.700
M_5	625	50.000				345.000	3	395.625
M_6	25.000						1	25.000
M_7		100					1	100
M_8		200.000					1	200.000
M_9		125.000					1	125.000
M_{10}		10.500					1	10.500
M_{11}		50.000		150.000	500.000	125.000	4	825.000
M_{12}			62.000			5.000	2	67.000
M_{13}		25.000	62.000				2	87.000
M_{14}		25.000				5.000	2	30.000
M_{15}	1.250						1	1.250
Sum	31.875	1.705.600	124.000	482.700	500.000	480.000	E Development P Beta test/introduction V Preproject W Real operation # geschätzte Größe	
Effort	35 PM	170 PM	160 PM	435 PM	145 PM	100 PM[#]		