R-LIME: improving the Risk dimension in the LIME model

Alain Abran[1], Luigi Buglione[2], Daniel Girard[3]

[1]*École de Technologie Supérieure – ETS, 1100 Notre-Dame Ouest, Montréal, Canada H3C 1K3*
aabran@ele.etsmtl.ca
[2] *École de Technologie Supérieure – ETS, 1100 Notre-Dame Ouest, Montréal,Canada H3C 1K3*
luigi.buglione@computer.org
[3] *Cisco Systems Canada Co., 1501, avenue McGill College, Bureau 600, Montréal, Canada H3A 3M8*
dagirard@cisco.com

Abstract. Risk management is gaining greater visibility in organizations, but is not always tightly integrated within Project Management: often risks are managed separately from the project plan, which is not revised accordingly, nor are revisions made to project estimates during the project lifetime. An open research issue is how to integrate risk evaluation into project (re)planning. In practice, project managers initiate the detailed planning process with the creation of a Gantt chart, first developing a Work Breakdown Structure (WBS). Recently, the Risk Breakdown Matrix (RBM) has been proposed to associate risk activities directly with the project tasks planned in WBS terms. In parallel, a model has been proposed to integrate three quantitative project perspectives concurrently, either at a particular time (the QEST model) or throughout the project life cycle (LIME). This QEST model was next generalized to an n-dimensional model. The integration of the RBM into LIME (to be referred to as the R-LIME model) is proposed here to allow quantitative determination of the risk associated with each project phase and to calculate the net performance value for the project, phase by phase. Key concepts of the R-LIME model are presented in this paper, including an example of its usage.

1. Introduction

"*Risk is the possibility of suffering loss*", according to Webster's Dictionary. In a software development project, these losses can take several forms, among them reduced delivered product quality to the Customer and increased production costs due to waste, rework, etc. This is also referred to as the Cost of Non Quality or the Price of Non Conformance. Van Scoy reports that: "*Risk in itself is not bad; risk is essential to progress, and failure is often a key part of learning. But we must learn to balance the possible negative consequences of risk against the potential benefits of its associated opportunity*" [15]. Risk Management (RM) can therefore be defined as "*the systematic process of identifying, analyzing, and responding to project risk. It* *includes maximizing the probability and consequences of positive events and minimizing the probability and consequences of adverse events to project objectives*" [12].

Risk management (RM) has been discussed in the software engineering community for some time, more recently through a tailored process in software process improvement (SPI) models: for instance, risk-related topics have been integrated into the Sw-CMM v1.1 [13] and CMMI v1.1 [14] models - Table 1

Table 1 – Risk-related items in software process improvement models

Maturity Level	Sw-CMM	CMMI
2	• SPP, Ac13 (*identification*) • SPTO, Ac10 (*tracking*)	• PP (*identification and planning*) • PMC (*monitoring*)
3	• ISM, Ac10 (*RM at the organizational level*)	• RSKM (*new PA expanded from the single Ac in ISM*) • DAR (*formal evaluation process to evaluate alternatives for selection and mitigation of identified risks*)
Legend	• SPP = Software Project Planning • SPTO = Software Project Tracking & Oversight • ISM = Integrated Software Management • RM = Requirement Management • Ac = *Activity*	• PP = Project Planning • PMC = Project Monitoring & Control • RSKM = Risk Management • DAR = Decision Analysis & Resolution • PA = *Process Area*

It is also to be noted that RM processes and practices are not fully integrated into the Project Management practices, but managed separately. For example, the British Computer Society has recently reported that *"[…] risk management is one of the most neglected aspects of IT project management. […] Regrettably, risk management is often limited to compilation of a risk register at the start of the project which plays little role in the day-to-day management of the project*" [2, p.26].

In this paper, we investigate how to integrate RM outcomes into iterative project re-estimation through an extension of the LIME model, this extension to be referred to as the **R-LIME** (Risk-LIME) model. Section 2 presents an overview of RM techniques and methods specific to the software sector, and proposes a taxonomy of such techniques. Section 3 illustrates a recent RM approach, the Risk Breakdown Matrix (RBM), and section 4 integrates it into the LIME model to improve project re-planning and re-estimation across the whole software project life cycle. Finally, section 5 presents some conclusions and suggestions for further research.

2. Risk Management models and approaches for the software sector

Since the mid ’80s, specific RM models and approaches have been tailored to the software sector on the basis of previous works developed in other business sectors, including insurance and banking[1]. An inventory from [7] is presented in Table 2 and Figure 1 by year of publication, together with an indication of whether each is based on a taxonomy of risks or on a process for managing risks.

Table 2 – Risk Management technique types listed in [7]

Year	Model/Approach	Author(s)	Type	Discipline	Comments
1989	---	Boehm	Taxonomy	Software Engineering	Three-tier levels (main phase; sub-phase; tools)
	DoD-Std-2176A	Charette	Taxonomy	Software Engineering	Identification of 6 main steps in risk management
1992	Guidebook Fundamentals?	Charette	Process	Software Engineering	The ethics of risk taking
	---	Chittister et al.	Process	Software Engineering	Risk Management Paradigm
	Indy Risk School	Thomsett	Process	Sw Programmer	Three-tier levels inside overall Project Management role
1995	International Standard 300-3-9	IEC	Process	Tech. Systems	Three-tier levels (main phase; sub-phase; tools)
1996	MAGERIT v1	MAP	Taxonomy	Software Engineering	Improvement of quality and productivity in information systems development process
	Continuous Risk Mgmt	Dorofee	Taxonomy	Software Engineering	Continuous Risk Management (CRM) Paradigm
	PMBOK version 1996	PMI	Process	Project Mgmt	Risk Management Module in the overall PMI methods (version 1996), chapter 11
1997	---	Soo Hoo	Taxonomy	Software Engineering	Gartner Group - Report on risk taking (DATAPRO - DISG)
1998	---	Girard	Process	Software Engineering	Risk Assessment in Technology Innovation projects
1999	---	Raz & Michael	Survey	Generic Project	Majority of Israeli risk projects are related to IT
2000	PMBOK version 2000	PMI	Process	Project Mgmt	Actualization of PMI methods (version 2000), chapter 11
2002	CMMI – Risk Mgmt PA	SEI	Process	Software Engineering	SW-CMM – 5 level phases with key process area focus
	………	Smith & Merritt	Process	R&D	Based on PMBOK & FMEA
2004	PMBOK version 2004	PMI	Process	Project Mgmt	Actualization of PMI methods (version 2004), chapter 11

Legend

Type	Comment – Definition
Taxonomy	Definition of a well-established taxonomy of risks
Process	Definition of a process for managing risk; no predefined taxonomy
Survey	Identification of most frequent risks in projects through questionnaires

[1] For example: **FERMA** (Federation of European Risk Management Associations – http://www.ferma-asso.org/6.html), **ARIA** (American Risk & Insurance Association – http://www.aria.org).

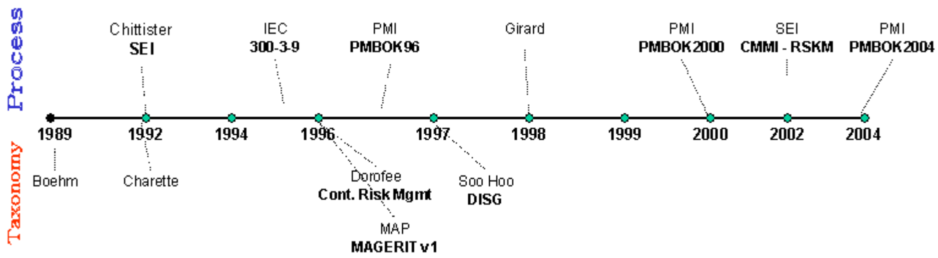


Fig. 1 – Evolution and classification of Risk Management methods in the software domain

3. RBM: the Risk Breakdown Matrix

In the Project Management domain, there is intensive usage of the WBS for planning purposes:

- **Work Breakdown Structure (WBS):** The WBS is the functional decomposition of project tasks. It is defined as "*a deliverable-oriented grouping of project elements that organizes and defines the total work scope of the project. Each descending level represents an increasingly detailed definition of the project work*." [12].

More recently, a mirror-like technique about risk, called RBS, has been introduced:

- **Risk Breakdown Structure (RBS):** Hillson derived the RBS from the WBS concept, with risk taxonomies substituted for project tasks. RBS is therefore defined as "*a source-oriented grouping of project risks that organizes and defines the total risk exposure of the project. Each descending level represents an increasingly detailed definition of sources of risks to the project*" [10]. As for the WBS, the use of three or four nested levels is suggested for detailing the risks. Hillson provides several examples of RBSs for distinct sectors, including software projects. It is expected that RBS will be included as a key concept in the 2005 edition of the PMBOK [11].

Subsequently, Grimaldi & Rafele combined the two techniques into a matrix called the *Risk Breakdown Matrix* (**RBM**) [9], with rows representing the WBS structure and the columns, the RBS structure. Each RBM cell will contain the amount of estimated risk (R) calculated in the usual way as the product of *probability* and *impact*:

$$Rwp_i = \sum_{j=1}^{n} P_{i,j} * M_{i,j}$$

where:
Rwp_i = risk value for the Ith Work Package
$P_{i,j}$ = probability of occurrence of the jth risk for the Ith Work Package **$M_{i,j}$** = impact due to the jth risk on the Ith Work Package

Table 4 provides an example of the RBM with two evaluations using this matrix: an evaluation ranked by the most risky work products (WPs - rows) and another by the most risky events (columns). Thus, RBM can be used as a risk analysis tool and as a tool for communication between the different roles active in a project, looking at risk evaluations using different levels of detail (e.g. granularity).

Table 3 – RBM structure

			RBS – risky events					**Evaluation by WP**	
			M1 $p_{i,1}$	M2 $p_{i,2}$	M3 $p_{i,3}$	**...**	Mn $p_{i,n}$	SR	**Rank by WP**
WBS Work Package	**WP1**	$I_{1,j}$						$\sum_j R_{1,j}$	
	WP2	$I_{2,j}$						…	
	WP3	$I_{3,j}$		$R_{3,2} = P_{2,3} * M_{3,2}$					
	WP4	$I_{4,j}$							
	WP5	$I_{5,j}$							
	…								
	WPm	$I_{m,j}$							
Evaluation by Risky Events	SR		$\sum_i R_{i,1}$	…					
	Rank by Risk type								

Three distinct types of ratings can be used when filling out such a matrix:

a) *impact* and *probability* are both rated in text form within a predefined ranking terminology scale (linguistic values); such ordered values can be sorted by criticality level, for instance (*ordinal* scale type);

b) *impact* and *probability* are both rated using a numerical scale (i.e. Likert scale) (*interval* scale type);

c) *impact* is rated against a parameter representing each single risky event, while *probability* as the % likelihood of occurrence of such an event.

Ratings on an interval scale are presented in the examples below, with impact and probability within a range from 0 to 10. Examples of equivalences of WBS and RBS elements at each level are presented in Table 4.

Table 4 – WBS and RBS equivalences by RBM level

RBM Level	WBS	RBS
0	Project (*root*)	Project risks (*root*)
1	Software Development Phase	Object for risk evaluation
2	Issue within a certain software development phase	Issue within a certain object for risk evaluation
3	Detailed task within the Sub-issue of a certain software development phase	Detailed risk within the Issue of a certain object for risk evaluation

Examples of candidate WBS and RBS taxonomies for managing a software development project are presented in the Appendices. The *level of depth* in the risk analysis is of interest to the analyst:

- On peer levels between WBS and RBS, [9] also defines a *risk pyramid* where each peer level has a related matrix: the higher the RBM level, the more general the analysis (at Level 1, the matrix obtained by crossing Level 1 WBS and RBS items analyzes the global project risk areas), the lower the RBM level, the more detailed (single task/issue level) the analysis (at Level 2, the matrix obtained detects the risks for project deliverables, and so on).
- On different levels between WBS and RBS: in this case, the analysis is to figure out how much a certain risk impacts a specific project phase, through single cells in the matrix (i.e. a specific project phase versus a specific group of risks). This is illustrated with an example in Table 5, with:
  - **WBS** – Project Management (Level 1): Planning, meeting and administration (Level 2)
  - **RBS** – Program constraints (Level 1): Resources, contract and program interfaces (Level 2)

Taking for granted that the values[2] collected at Level 2 for a specific instantiation are as listed in Table 5, the R value for each cell (that is, the risk value for the I^{th} Work Package) is calculated as the *probability by impact* (that is, $Rwp_i = \sum_{j=1}^{n} P_{i,j} * M_{i,j}$).

Table 5 – RBM example (Level 2 excerpt): program constraints risks in PM activities

		RBS (from Program Constraints)			Evaluation by WP		
	Level 2	**Resources**	**Contract**	**Prg Interfaces**	**SR**	**%**	**Rank by WP**
WBS (From Project Mgmt)	**Planning**	R=199	R=109	R=51	**359**	**63%**	**1**
	Meeting	R=35	R=6	R=6	**47**	**8%**	**3**
	Administration	R=48	R=15	R=99	**162**	**29%**	**2**
Evaluation by Risky Events	**SR**	**282**	**130**	**156**	**568**	**100%**	
	%	**50%**	**23%**	**27%**	**100%**		
	Rank by Risk type	**1**	**3**	**2**			

Thus, the total risk value is equal to **568**, with the most risky PM activity (from WBS) being Planning (**63%**), with the most risky external constraint element (from RBS) being Resources (**50%**). For a better understanding of what the risky aspect is in Planning or in resource management, RBM Level 3 must be analyzed next, and so on.

[2] $\mathbf{P_{i,j}}$ = probability of occurrence of the j^{th} risk for the I^{th} Work Package and $\mathbf{M_{i,j}}$ = impact due to the j^{th} risk on the I^{th} Work Package

4. R-LIME: improving the LIME model with RBM

In this section, we address the following issue: how to do we integrate the information from the RBM into the re-planning of the project from phase to phase?

4.1. The QEST model

The QEST model allows for the integrated measurement of project performance as measured at a detailed level from multiple viewpoints [3]. Graphically, it is represented in 3D with a regular tetrahedron (Figure 2), using its original three perspectives: Economic, Social and Technical (E, S, T); this version of the model is referred to as the '**QEST 3D**'. The overall project performance (p) is determined using the corresponding classic geometrical formulae, such as the volume of a truncated tetrahedron defined by the individual perspective values (Qe, Qs and Qt), divided by the whole tetrahedron volume.

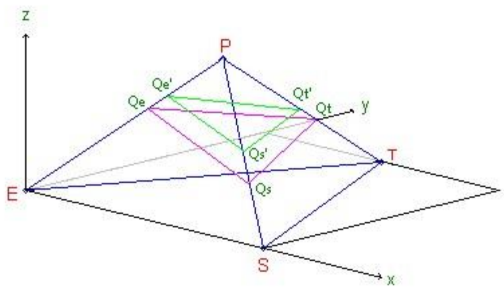


Fig. 2 – The QEST model

The geometrical foundations for the model are documented in [4], including a discussion for selecting the volume as the preferred geometrical concept for measuring performance. This is in lieu of distance and area, which represent only partial views of project performance.

4.2. The LIME model

To measure the performance of a software project across its successive project life cycle phases, Buglione & Abran proposed the **LIME** (LIfe cycle MEasurement) model [5] (Figure 3).

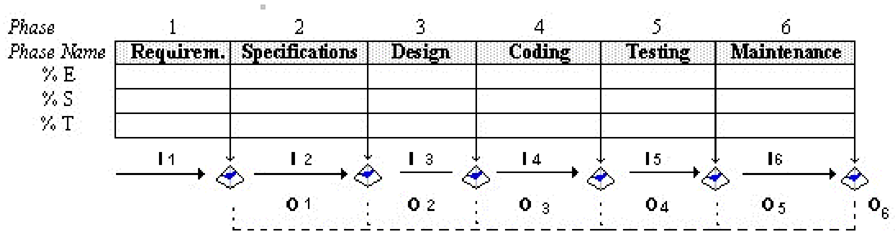


Fig. 3 – The LIME model for the software life cycle (SLC) phases

In this model extension, the output of the (n-1)th phase represents the input for the nth one, and so on. LIME is used for consolidating data from each software life cycle (SLC) phase and for determining *ex-post* the proper process improvements within projects. This kind of relationship, described between subsequent SLC phases, suggests other usages of LIME for planning purposes, using the historical series of data gathered from completed projects:

- **Identification of lead indicators from trend analysis of past projects**: for instance, decreasing average values between phases *n* and *(n+1)* suggests an investigation of what happened in phase n (methods, resources, tools, external constraints, etc.) to identify corrective actions in a new project with similar characteristics. This, of course, would require a Root-Cause Analysis (**RCA**) to analyze the gaps between expected and recorded SLC phase performance values, with the object of working out at what specific time in the project a problem can initially cause failures and faults.
- **Estimation models**: construction, through regression analysis, of estimation models for optimizing the amount of resources to use in the next SLC phase, and therefore fine-tuning the overall project estimates (people effort and other proxies like defects, requirements, etc.). This can be done in an organization with a historical database of the QEST indicator composition for each individual project and then selecting performance values from a set of similar projects. At least two levels of granularity in performance can be investigated:
  - **Project level**: the estimation model will consider the overall *p* value (per *project* using the QEST model; per *SLC phase* using the LIME model);
  - **Perspective level**: the estimation model will consider the single *p* values for each of the perspectives considered (per *project* using QEST model; per *SLC phase* using the LIME model).

4.3. Performance model extensions

The initial QEST 3D model was generalized next in [6] to handle any number of concurrent viewpoints in the n-dimensional space – **QEST nD,** allowing organizations to select and handle any number of viewpoints selected to be taken into consideration at any specific time.

QEST nD can therefore be used as a general-purpose multidimensional measurement model, whatever the application domain. This has been illustrated, for example, in the joint usage of the QEST model and the ICT Balanced Scorecard (BSC) [1], using the QEST nD model within each BSC perspective, as well as with the integration of all the BSC perspectives.

4.4. R-LIME: the Risk dimension extension

Gotterbarn has suggested the possibility of using the LIME model from a specific viewpoint of performance, that is, from a risk viewpoint [8]: he pointed out that it could handle a partial and implicit risk evaluation and rating, with the concurrent presence of several groups of stakeholders in evaluating a project's performance. We have implemented his suggestion in this proposed extension to the LIME model for Risk Management. LIME had been initially designed as a model for ***ex-post*** analysis through consolidation of performance evaluations from multiple concurrent viewpoints. It could also, of course, be used for defining performance targets through the SLC phases: taking risks into consideration can help improve the estimation process. In order to do this, we have investigated the use of **RBM** and discuss it next.

It is important to understand that risks should be monitored and managed continuously by considering the amount/level of risk expected between consecutive SLC phases, and later mitigated during the life cycle. Throughout the life cycle, risks can influence the ***p*** performance values, that is, there is a relationship between risks, estimates and performance, as illustrated in Figure 4:

- *What kind of relationship exists between SLC phase performances and risks in each phase?* Risk should be taken into account when making estimates. The greater the gap between the estimates and the assessment of risks, the greater the (re)planning and estimation capability of the Project Manager during the project life cycle. These relationships are summarized in Figure 4.

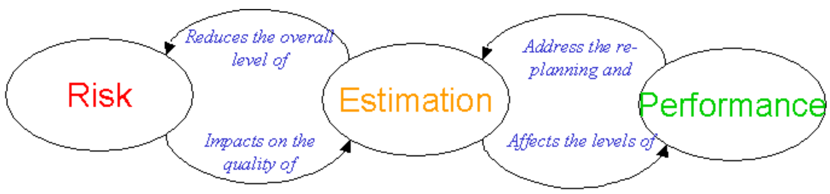


Fig. 4 – Relationships between Risk, Estimation and Performance

- *How are risk assessment and performance values to be related?* Considering the RBS taxonomy from the RBM, each risk (e.g. tester expertise) must be linked to a specific SLC phase (testing, in this example), to the indicators chosen for that QEST phase set of indicators and the Δ% discounted from the values calculated. Repeating this for all risks considered in the RBS per SLC phase, a new $\boldsymbol{p}_{\mathbf{r})}$ value (of performance) can be derived taking risks into consideration – see Figure 5.

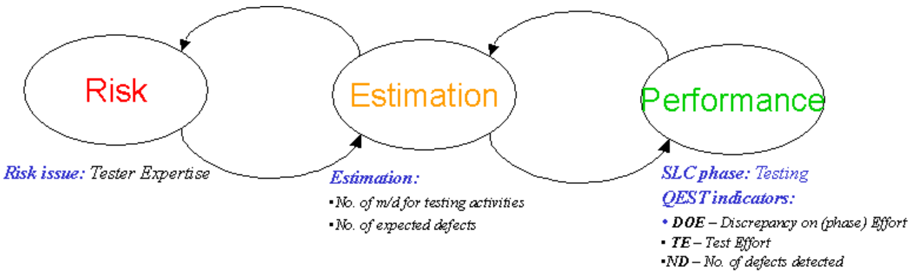
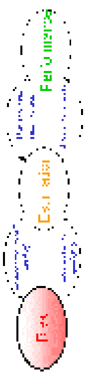


Fig. 5 – An example of a causal relationship

- *What is the appropriate time for execution of a revised performance calculation?* At the end of each SLC phase, the results obtained in the phase review meeting can be used for re-estimating resources for the next project phase, on the basis, of course, of a number of parameters.

4.5. R-LIME: an example

An example involving tester experience and staff is discussed next, using the RBM matrix excerpt presented in Figure 6: the "**Testing**" row (WBS Level 1) and the "**Staff**" column (RBS Level 3) are highlighted in bold.



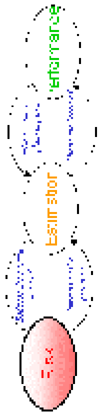
		RBS				
		Program Constraints				
		Resources				
		Staff	Budget	Facilities	R	R%
WBS	Analysis	20	25	60	**105**	28
	Design	15	30	10	**55**	15
	Coding	18	20	25	**63**	17
	Testing	30	35	15	**80**	**21**
	Maintenance	25	25	25	**75**	20
	R	**108**	**135**	**135**	**378**	**100**
	R%	**29**	36	36	**100**	

Fig. 6 – RBM matrix (2^{nd} level): an example

In the example in Figure 6, the risk contribution of testing activities contributes to 21% of the total risk; the share of risk related to staffing is 29% of the "Resources" risks, and this is the risk with the lowest incidence in that Level 3 group. The focus can now be put on the "staff" risk within the "Testing" activities. The hypothesis is that, after a risk review (Figure 7), the initial evaluation of an R=30 has been reduced to R'=10 (a risk mitigation action might have been to involve three senior testers in place of the five junior testers initially planned). At reassessment time, the risk on "Staff" had been decreased by 67%, with a consolidated impact reduced by 5%

(Figure 7). The example in Figure 7 also includes three indicators, which have been linked to the “staff” risks[3] in this project:

1. Project Delivery Rate (**PDR**) = Function Points / Work Effort;
2. Duration Delivery Rate (**DDR**) = Function Points / Elapsed Time and
3. Delivery Defect Density (**DD**) = Function Points / Defect Density.



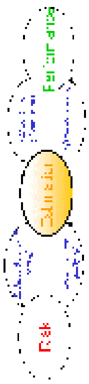
			RBS – Program Constraints – Resources: Staff	Budget	Facilities	R	R%	Staff R%
WBS	Testing	R	30	35	15	80	21	8
		R'	10	24	25	59	17	3
		%Diff.	-67	-31	67	-26	-22	-5
		Rel-Ind						

PDR (FP/WE) - Project Delivery Rate
DD (FP/ND) - Inverse of Defect Density
DDR (FP/ET) - Duration Delivery Rate
...

recalculated on R=358 (378-20)

Fig. 7 – *Risk: new staff R% values after risk reassessment*

In the context of a relationship between Risk/Estimation/Performance, a difference in the risk assessment has an impact on the estimations for some indicators. The example in Figure 8 illustrates how a reduction in risks on the “staff” driver of the 67% can lead to a modified estimate (from 36 days in the initial estimate to a revised estimate of 12 man-days). Obviously, those hypotheses should be discussed and verified against historical data (where available) and/or brainstorming sessions within the project team. Figure 8 shows the new values for the three indicators in the example in Figure 7.



Testing SLC phase	*Aspects*	Initial Estimation	Risk assessment (%)	Estimation revised
	Effort	36 m/d	-66,67 to be applied on the indicators related to that risky event (**PDR**)	**12 m/d**
	Duration	12 calendar days	**DDR**	**4 calendar days**
	Defectability	10 defects	**DD**	**5 defects**

Hp:
Testing activities must be conducted by 3 FTE (full-time equivalent) people

Fig. 8 – Estimation: the discounting phase measures initial estimated values

[3] QEST/LIME are *open models*, with a non-predefined set of indicators. Indicators must be chosen for each perspective.

The final step involves the "translation" and usage of such new values for recalculating the new ***p*** value, shown as $\boldsymbol{p_r}$ in Figure 9. This requires a rerun of the QEST calculation using the same initial data and substituting only the reassessed values impacting the measures calculated in Figure 7.

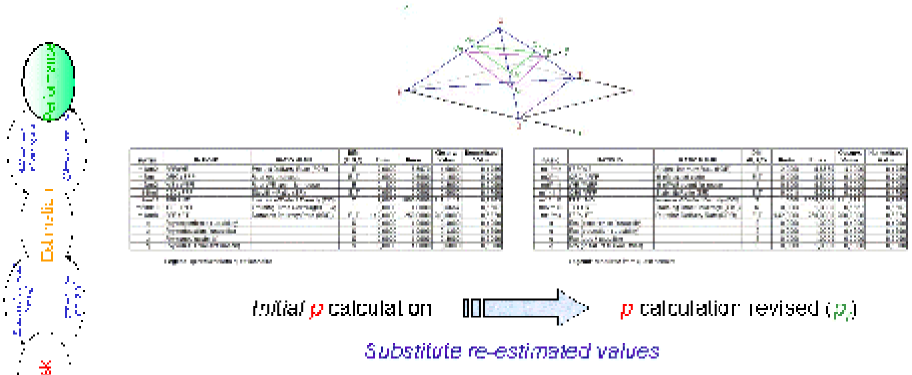


Fig. 9 – Performance: recalculation of p values (delete hyphen in "re-estimation" above)

Supposing that the SLC Testing *p* value were $p=0.7$, and that, after these modifications have been made, the new revised value were $p_r=0.75$, with better performance (+5%) the following comments could be made:

- The resources assigned to testing activities had the right set of skills, and the amount of risk impacting their effort estimation and schedule was too high. Possible candidate improvement actions could be:
  - Skill inventory detail
  - Cost figures per professional
  - Historical data on average productivity figures from projects segregated by SLC phase and average number of people involved in each SLC phase

5. Conclusions & Prospects

The introduction of best practices surrounding risk management is becoming increasingly important for organizations. Until recently, the emphasis in project planning had been more on the identification of risks than on the quantification of "how much" risk, since the impact could only be managed as a qualitative proxy.

To quantitatively manage project risks, the RBM (Risk Breakdown Matrix) technique has been proposed to provide numerical values for risks and to link them to related tasks using the project's WBS. To use the RBM throughout the many project phases, the RBM technique was integrated into the LIME model, a model for determining the performance of a software project through its life cycle phases. Taking into account the LIME model, we considered the "QEST nD" model as a basis for each SLC phase, in order to apply *n* possible viewpoints in the calculation of project performance.

After describing the relationships among risk, estimation and performance in projects, a revised version of the LIME model for Risk called **R-LIME** was presented, and examples were included of how a performance result could be used for improvement actions within the project. Use of the R-LIME model would make this possible, through the evaluation of risks on single WBS items (or groups of items), to share and use them with the project indicators, thereby deriving further information for project monitoring and control during the project lifetime, as requested in most SPI models.

Further evolutions of R-LIME will be investigated next, including:

- A more extensive simulation using the International Software Benchmarking Standards Group – ISBSG – data repository for illustrating case studies with industry data applying the model and for deriving further information about relevant linkages in software projects;
- The derivation of estimation models for QEST/LIME using project data from the ISBSG repository.

References

1. Abran, A., Buglione L.: A Multidimensional Performance Model for Consolidating Balanced Scorecards, International Journal of Advances in Engineering Software, Vol. 34, No. 6. Elsevier Science Publisher (2003) 339-349
2. BCS: The Challenge of Complex IT Projects, Technical Report (2004). Royal Academy of Engineering & British Computer Society. ISBN 1-903496-15-2. URL: http://www1.bcs.org.uk/DocsRepository/07500/7560/complexity.pdf
3. Buglione, L., Abran A.: Multidimensional Software Performance Measurement Models: A Tetrahedron-based Design. In Dumke R., Abran, A. (eds.): Software Measurement: Current Trends in Research and Practice. Deutscher Universitats Verlag GmbH, ISBN 3-8244-6876-X (1999). 93-107
4. Buglione, L., Abran A.: Geometrical and statistical foundations of a three-dimensional model of software performance. International Journal of Advances in Engineering Software, Vol. 30, No. 12. Elsevier Science Publisher (1999). 913-919
5. Buglione, L., Abran A.: LIME: A Three-Dimensional Software Performance Measurement Model for Project Management, In: Proceedings of the 2WCSQ (2nd World Congress for Software Quality), Yokohama (Japan), 25-29 September 2000. 163-168, URL: http://www.lrgl.uqam.ca/publications/pdf/559.pdf
6. Buglione, L., Abran A.: QEST nD: n-dimensional extension and generalisation of a Software Performance Measurement Model. International Journal of Advances in Engineering Software, Vol. 33, No. 1. Elsevier Science Publisher (2002). 1-7
7. Girard, D.: Évaluation de risque de projet d'innovation technologique, Rapport Final DGA110. Programme de doctorat en genie, École de Technologie Superieure, Université du Québec à Montréal (2003)
8. Gotterbarn, D.: Reducing Software Failures: Addressing the Ethical Risks of the Software Development Lifecycle, Australian Journal of Information Systems (AJIS), Vol.9 No.2. University of Wollongong (2002). URL: http://www.sdresearch.org/Software%20Tools/Reducing_software_failures.pdf

9. Grimaldi,S., Rafele C.: Analisi del Rischio. Modello Gerarchico per l'Analisi del Rischio, De Qualitate, Ed. Nuovo Studio Tecna (2004). 22-38
10. Hillson, D.: Using a Risk Breakdown Structure in project management, Journal of Facilities Management, Vol.2. No.1. Henry Stewart Publications (2003). 85-97, URL: http://www.risk-doctor.com/pdf-files/rbs0603.pdf
11. PMI: A Guide to the Project Management Body of Knowledge (PMBOK). Project Management Institute (2000). ISBN 1-880410-25-7
12. PMI: Practice Standard for Work Breakdown Structure. Project Management Institute, (2001). ISBN 1-88410-81-8
13. Paulk, M.C., Weber, C.V., Garcia, S.M., Chrissis, M.B., Bush, M.: Key Practices of the Capability Maturity Model Version 1.1, CMU/SEI-93-TR-025, Technical Report. Software Engineering Institute/Carnegie Mellon University (1993). URL: http://www.sei.cmu.edu/pub/documents/93.reports/pdf/tr25.93.pdf
14. CMMI Product Team: CMMI for Software Engineering, Version 1.1, CMMI-SE/SW/IPPD/SS v1.1, Continuous Representation, CMU/SEI-2002-TR-011, Technical Report, Software Engineering Institute (2002), URL: http://www.sei.cmu.edu/pub/documents/02.reports/pdf/02tr011.pdf
15. Van Scoy, R.L.: Software Development Risk: Opportunity, Not Problem. CMU/SEI-92-TR-30, Technical Report. Software Engineering Institute (1992). URL: http://www.sei.cmu.edu/pub/documents/92.reports/pdf/tr30.92.pdf

Appendix A: *WBS for software development* [9]

Level 0	Level 1	Level 2	Level 3
WBS for a Software Development Project	Project Management	Planning	Develop the Project Plan
			Define the Scope
			Develop the Resource Plan
			Develop the Communication Plan
			Develop the Risk Plan
			Develop Modification Control Plan
			Develop the Quality Plan
			Develop the Purchasing Plan
			Develop the Cost Plan
			Develop the Organizational Plan
			Develop the Project Program
		Meeting	Kick-off Meeting management
			Weekly tracking meetings
			Monthly tactical meetings
			Project Closure meeting
		Administration	Standards
			Program control
	User Requirements (*Analysis*)	Product Requirements	Software Requirement draft doc
			Verification of Sw Req draft doc
			Updating of Sw Req draft doc
			Final Verification of Sw Reqs
			Approval of Sw Requirements
		User Manual	User Manual Draft document
			Verification of User Manual draft doc
			Updating of User Manual draft doc
			Final Verification of User Manual
			Approval of User Manual
		Training programs	Definition of the Training Programs
			Verification & Approval of TPs
			Definition of materials for training
			Verification & Approval for materials
			Delivery of materials for managing training
			Updating of Training Programs
		Hardware	Hardware Requirement draft doc
			Verification of Hw Reqs doc
			Approval of Hw Reqs doc
	Software Specifications (*Design*)	Definition of initial Sw Project	
		Verification of initial Sw Project	
		Update of initial Sw Project	
		Final Verification of Sw Project	
		Approval of Sw Project	
	System Construction	Software Configuration Management	
		User Manual tailorings	
		Training materials tailorings	
		Hardware installation	
		Implementations and future support	
	Sofware Testing & Integrations	Software Coding	
		System Test Plan	
		System Test Cases	
		System Test Results	
		User Manual	
		Training Materials	
		Hardware	
		Implementation and Future support	

Appendix B: *RBS for software development* [10]

Level 0	Level 1	Level 2	Level 3
Software Project Risk	Product Engineering	Requirements	Stability
			Completeness
			Feasibility
			…
		Design	Functionality
			Interfaces
			Testability
			…
		Code & Unit Test	Feasibility
			Testing
			Coding / Implementation
			…
		Integration Test	Environment
			Product
			System
			…
		Engineering specialties	Maintainability
			Reliability
			Security
			…
	Development Environment	Development process	Formality
			Process control
			Product control
			…
		Development system	Capacity
			Reliability
			System Support
			…
		Management process	Planning
			Project Organization
			Management experience
			…
		Management methods	Monitoring
			Configuration Management
			Quality Assurance
			…
		Work Environment	Cooperation
			Communication
			Morale
			…
	Program Constraints	Resources	Staff
			Budget
			Facilities
			…
		Contract	Type of contract
			Restrictions
			Dependencies
			…
		Program interfaces	Customer
			Subcontractors
			Corporate Management
			…