Journal of Theoretical and Applied Computer Science
ISSN 2299-2634
Vol. 6, No. 2, 2012, pp. 13-26
http://www.jtacs.org

Towards expert-based modelling of integrated software quality

Łukasz Radliński
University of Szczecin, Faculty of Economics and Management
lukasz@radlinski.edu.pl

Abstract: *This paper reports on a part of a project aimed at building an probabilistic model for integrated software quality simulation and prediction. This paper discusses results of the questionnaire survey focused on gathering expert knowledge about the factors influencing various features of software quality. Specifically, this analysis identifies project and process factors of software quality, investigates relationships between quality features and their sub-features as well as priorities for quality features. The survey has been performed among software engineering experts and projects managers. Obtained results will be used to calibrate that model for software quality simulation and prediction. These results also partially deliver a general overview on how software quality features are perceived by industry.*

Keywords: *software quality, software process, expert knowledge, survey study, data analysis*

1. Introduction

Quality is one of the main drivers in software projects. Achieving high level of software quality is difficult without appropriate management activities that require certain inputs. These inputs may have various sources, such as expert judgment, process data or software metrics. They may be combined in models to enable reuse of existing knowledge in different projects. Such models can be used for estimation, simulation and prediction of software quality and thus extend the base for decision support.

Quality prediction models have been built since the turn of 1960's and 1970's. They involved using a range of techniques such as regression, neural networks, decision trees, support vector machines, case-based reasoning [17]. A common feature of these models is that they are typically focused on a single attribute of quality, such as number of defects, defect-proneness, reliability, security, usability, maintainability, etc. [6][7][8]. Very few models incorporate multiple software quality features.

The aim of an on-going research project is to develop a model that could be used for simulation and prediction of integrated software quality. In this context the term 'integration' refers to capturing a variety of quality features, linked with each other and with a set of influential factors, in a single model. Earlier analyses of empirical data and attempts to build a simulation and predictive model have been published in [10][11][13][14].

The aim of this paper is to analyze the results of surveys that have been performed among experienced project managers and software engineering experts. The goal of these surveys was to gather opinions on factors that influence different quality features. The results of this analysis serve as one of the sources for developed simulation model.

The paper is organized as follows: Section 2 discusses the background and motivation for this study. Section 3 considers the research approach by explaining the procedure, summary of techniques used and the questionnaire design. Section 4 provides the details of the results of the analysis. Section 5 discusses lessons learned and threats to validity of achieved results. Section 6 provides conclusions and ideas for future work.

2. Background and motivation

The aim of the surveys among software engineering experts and project managers was to gather personal knowledge on the factors influencing quality features. These surveys have been performed to calibrate a Bayesian network model for software quality prediction and simulation. The core structure of that model has been defined in advance. Figure 1 illustrates the schematic of this model. The model has a modular structure, i.e., variables are grouped into topical subnets:

- Project factors – contains various factors describing the nature of the project and its environment, i.e. *architecture*, *CASE tool usage*, *deployment platform*, *user interface type*, *target market*, *used methodology*, *project difficulty*. These project factors influence selected quality features.
- Process factors – contains various factors describing the quality of development process. Depending on particular version of the model, this subnet may be more or less complex. In the smallest version it contains only very few details – *effort* and *overall process quality*, separately for each of three main development activities: specification, development and testing. In the most complex version it contains detailed process and people factors as discussed in Section 4.5 and illustrated in Figure 6. These process factors influence selected quality features.
- Quality features – contains a set of interconnected high-level features reflecting software quality.
- Quality features, sub-features and measures – contains a hierarchy of software quality where features are decomposed into a set of sub-features, and sub-features may have detailed quantitative measures assigned. This hierarchy is based on an ISO 25010 standard [1].
- Integration of components – enables reflecting integration of software components into larger artifacts such as sub-systems and systems and thus modeling the level of aggregated quality features.

It is beyond the scope of this paper to discuss the details of this model. Interested readers may find the details in [10][11][13].

The motivation for developing this model is the fact that we have not found another model aimed at predicting/simulating such variety of factors. Typically existing models are focused on a single feature of quality, such as maintainability [15] or defects [1][3][4]. After extensive literature survey we found only two studies closely relevant to the one that we have been developing.

The first of them [2], explicitly refers to the ISO 9126 standard, the predecessor of ISO 25010 standard that we have been using. However, there are two main problems with proposed model. The author does not provide details on the quantitative definition of that model, i.e. probability distributions. Thus, it is difficult to validate and reuse that model. Additionally, it was developed based on the data from small student projects. Therefore, it may be out of scope for larger industry-scale projects.

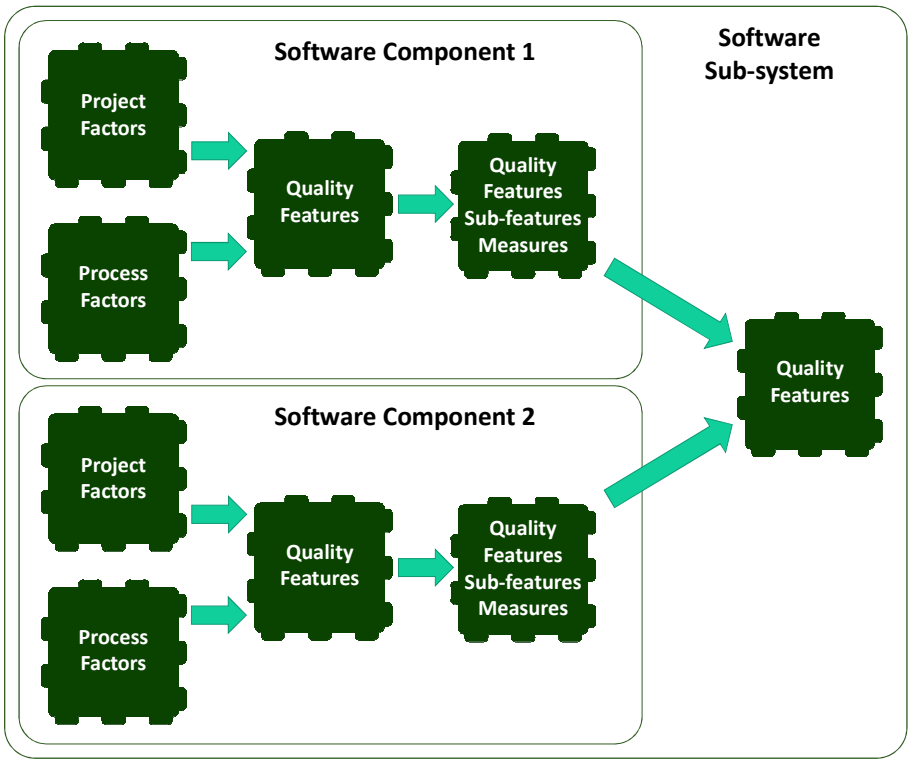

Figure 1. Schematic of the Integrated Model for Software Quality Prediction

The second study [16] was focused on developing a framework for building Bayesian networks for software quality prediction. However, it is not clear if a proposed framework can be effectively used to build models for integrated software quality prediction, i.e. where quality features depend on other factors but also on each other, as is the case in our study.

3. Research approach

3.1. Research procedure

As mentioned earlier, the questionnaire survey is a part of a larger work aimed at developing an integrated model for software quality prediction and simulation. The main stages of this work involve the following steps:

1. Defining the core structure of the model – results published in [10][11][13].
2. Gathering empirical data to partially calibrate the model – partial results published in [12][14].
3. Gathering subjective expert knowledge to partially calibrate the model – this is the core of this study.
4. Calibrating the model using a combination of results from steps 2 and 3 – future work.
5. Validating the model – future work.

To gather subjective expert knowledge we performed a questionnaire survey among software engineering experts and project/team managers in software projects. The surveys have been performed as direct interviews. We choose not to perform such survey by asking respondents to fill out questionnaire on-line for two main reasons. First, when performing a survey to calibrate an earlier model, we observed that some respondents provided responses without sufficient focus and understanding of questions and answers. Second, although the

core questionnaire was a formalized document we wanted respondents to give an ability to verbally provide additional information that was impossible to be included in the questionnaire. To support this, some interviews were recorded and during other we were taking live notes.

During the interview we presented five different versions of the model for integrated software quality prediction. The differences between models were related to the level of details of particular sub-networks. The aim of this step was to briefly familiarize respondents with the objective and the background of the interview. Then, during the main part of the interview we asked respondents to fill our questionnaire forms, one-by-one. The average duration of an interview was about 1:55 hours, the shortest took 1:25 hours and the longest 2:17 hours. The details of this questionnaire forms are discussed in Section 3.2.

We performed the interviews with eight selected respondents from six companies, of which four are Polish branches of major international IT companies, one an IT department of a large Polish bank, and one a systems development department of major electronics supplier for automotive industry based in Germany. The subjects for these interviews were selected based on their knowledge and experience in software development, in particular in software quality. Some of them participated in earlier studies on software validation and verification or calibration of earlier Bayesian network models.

After performing the interviews, we aggregated gathered data and performed cleaning. At this stage we corrected obvious mistakes that even some respondents noticed during an interview, for example the direction of the influence of particular factor on another one. Then we performed data analysis that involved variety of analytical techniques, such as basic measures of central tendency and variability, Spearman's rank correlation coefficient [9]. In addition, the analysis involved visual techniques, such as histograms, box-plots, scatter-plots as well as other custom graphs. The results of these analyses will serve as the input to the developed simulation model formally represented as a Bayesian network.

3.2. Questionnaire design

The main part of the questionnaire consists of seven groups of questions:

1. Importance of quality features – reflecting respondent's opinion on the priority for each quality feature expressed on a scale [0, 10].
2. Relationships between quality features – reflecting respondent's opinion on how strong are quality features related with each other. The lowest value '-5' indicates strong negative relationship, the value '0' – no relationship, and the highest value '+5' – strong positive relationship.
3. Hierarchy of quality features – reflecting respondent's opinion on the strength of relationships between each feature and a set of its individual sub-features. The same scale as in point 2.
4. Strength of impact of development process on quality features – reflecting respondent's opinion on how each of the main processes, i.e. specification, development and testing, influences quality features. The same scale as in point 2.
5. Relationships among detailed process factors – reflecting respondent's opinion on the factors that influence the aggregated process quality, separately for specification, development and testing. The same scale as in point 2.
6. Impact of project factors on quality features – reflecting respondent's opinion on the presence (indicated by entering a '+' sign) or absence (a '–' sign) of the impact of seven predefined project factors on quality features.

7. Strength of impact of project factors on quality features – reflecting respondent's opinion on the strength of impact of seven predefined project factors on quality features. This is an extension of point 6. The strength is expressed also on a scale [-5, 5]. But here the scale has a different interpretation. A value '-5' indicates that with a presence of specific project factor a given quality feature is expected to have a very low level. A value '+5' indicates that with a presence of specific project factor a given quality feature is expected to have a very high level.

Respondents were asked to provide answers as integer numbers. However, they were also informed that they may provide answers as intermediate values, e.g. '3.5', or as ranges, e.g. [2-3] or [-2, 3].

4. Results

4.1. Importance of quality features

To set up the focus of the interview on those quality features that are important from respondent's perspective, the first question of the main part of questionnaire was related to rating each quality feature depending on their importance. Figure 1 illustrates the values of the weighted means for importance of each quality feature according to respondents' opinions. The most important quality features appear to be *functional suitability*, *performance efficiency*, *usability*, *reliability*, and *security*. The least important features seem to be: *transferability* and *context coverage*.

Even though the group of respondents was small, we can observe high variability between provided answers for most quality features. The highest variability can be observed for *transferability*, *effectiveness* and *satisfaction*, while the lowest for *performance efficiency* and *freedom from risk*.

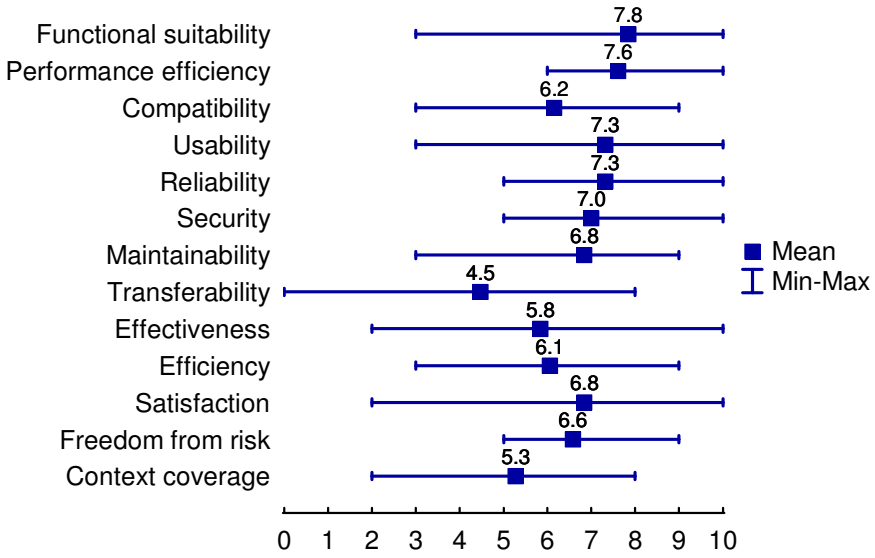

Figure 2. Priorities for quality features

Quality features	Functional suitability	Performance efficiency	Compatibility	Usability	Reliability	Security	Maintainability	Transferability	Effectiveness	Efficiency	Satisfaction	Freedom from risk	Context coverage
Functional suitability						0.48	0.47						
Performance efficiency					0.77		0.68				-0.70	0.74	
Compatibility				-0.79		-0.66			-0.81	-0.79	-0.55		
Usability			-0.79			0.51		-0.72	0.94	0.92	0.71	-0.51	0.48
Reliability		0.77										0.81	
Security	0.48		-0.66	0.51				-0.58		0.59			
Maintainability	0.47	0.68							-0.55				
Transferability				-0.72		-0.58			-0.61	-0.83			-0.68
Effectiveness			-0.81	0.94			-0.55	-0.61		0.90	0.71		
Efficiency			-0.79	0.92		0.59		-0.83	0.90		0.66		0.51
Satisfaction		-0.70	-0.55	0.71					0.71	0.66			0.50
Freedom from risk		0.74		-0.51	0.81								-0.64
Context coverage				0.48				-0.68		0.51	0.50	-0.64	

Figure 3. Spearman' correlations between priorities of quality features

Based on obtained values of priorities for each quality feature, we investigated the correlations between these priorities. Figure 3 illustrates the values of Spearman's rank correlation coefficient statistically significant at $p<0.05$. The strongest positive correlations can be found between priorities for the following pairs: *usability–effectiveness*, *usability–efficiency* and *effectiveness–efficiency*, while the strongest negative for pairs: *efficiency–transferability*, *compatibility–effectiveness*, *compatibility–efficiency*, and *compatibility–usability*. These negative correlations indicate the trade-offs in priorities between pairs of quality features, i.e. that with an increase of the importance for one feature we should expect a decrease of importance for the second feature.

4.2. Relationships between quality features

Figure 4 illustrates direct relationships between quality features based on answers provided by the respondents. The questionnaire did not contain questions on the causal direction of relationships within pairs of features, thus the results provided in this figure are symmetrical. The darkness of each cell indicates the frequency with which a given pair of quality features was pointed by respondents as being related with each other. The most frequently pointed relationships are for the following pairs: *functional suitability–satisfaction*, *functional suitability–usability* and *maintainability–transferability*. Most pairs of quality features have been pointed as related with each other by at least one respondent. However, there are 25 pairs that have not been pointed by any respondent (indicated with white background in Figure 4).

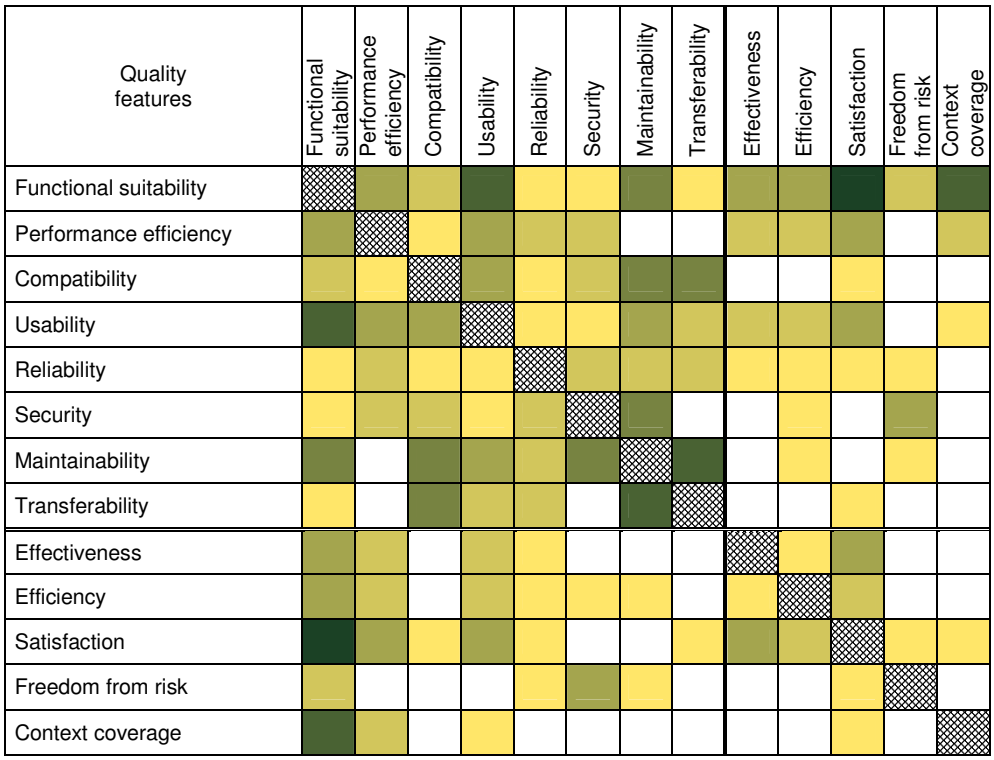

Figure 4. Relationships between quality features (darker color indicates stronger impact)

4.3. Hierarchy of quality features

In the next questions we asked questions related to the hierarchy of quality features, where main level features are decomposed into a set of detailed sub-features. Obtained answers provide information on the importance of particular sub-features for overall features. Figure 5 illustrates this hierarchy for almost all features. Two features, *effectiveness* and *efficiency*, are not listed there because each of them has only one sub-feature named exactly the same as its parent, i.e. main level feature. We note that, even though all sub-features have been illustrated on the same graph, the ratings of sub-features that describe two different features are not comparable. The aim of this question was to assess how important are different sub-features for a specific feature.

For *functional suitability* all three sub-features seem to be similarly important, with *functional completeness* only slightly less important than other two remaining sub-features.

For *performance efficiency* a sub-feature *time behaviour* is significantly more important than other two sub-features. However, this sub-feature has a high variability – for some respondents *capacity* was a more important factor for *performance efficiency* than *time behaviour*.

Interoperability seems to be a more important factor of *compatibility* than *co-existence*.

For *usability* three sub-features seem to be dominant: *appropriateness recognizability*, *operability*, and *user interface aesthetics*. *Accessibility* is moderately important, while *learnability* and *user error protection* are the least important.

For *reliability* the most important sub-feature is *recoverability*, followed by *availability*, while the least important is *fault tolerance*.

Integrity is the most important sub-feature for *security* and the least important are *non-repudiation* and *accountability*.

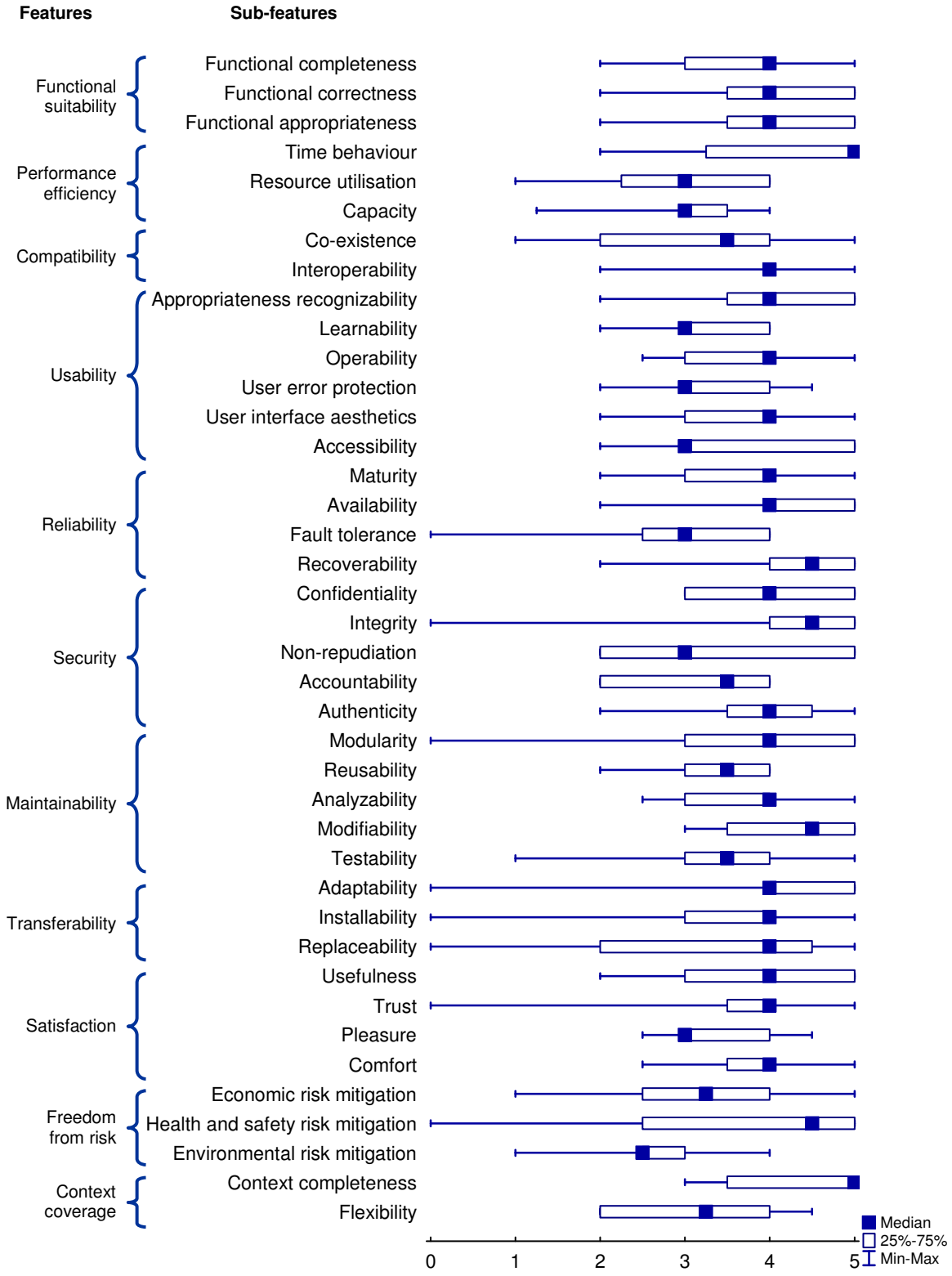

Figure 5. Hierarchy of quality features

For *maintainability* the most important are *modifiability* and *modularity*, the least important are *reusability* and *testability*, but the differences between all these sub-features are very narrow.

For *transferability* all three its sub-features are on the similar level. Interestingly, all three sub-features have very high variability and cover the whole range [0, 5] of possible values.

Usefulness seems to be the most important factor for *satisfaction*, and *pleasure* the least important.

For *freedom from risk* the most important seems to be *health and safety risk mitigation* and the least important – *environmental risk mitigation*.

For *context coverage* a sub-feature *context completeness* seems to be significantly more important than *flexibility*.

4.4. Impact of development process on quality features

Figure 6 illustrates the strength of impact of specification, development and testing processes on quality features. For only three pairs of variables the median value of '5', i.e. indicating the highest possible impact, were found: *specification process* → *functional suitability*, *development process* → *performance efficiency* and *development process* → *maintainability*. The lowest impact could be found for *specification process* → *maintainability*, *development process* → *functional suitability*, *testing process* → *maintainability*, and *testing process* → *freedom from risk*. Similarly to the priorities of quality features investigated earlier, also the level of impact of process factors on software quality has high variability indicated by wide ranges between 25th and 75th percentiles.

Respondents clearly noticed the differences of the impact of development activities on particular features. For example, while *specification process* and *testing process* have high impact on *functional suitability*, the *development process* has very weak impact. Conversely, while *specification process* and *testing process* have weak impact on *maintainability*, the *development process* has very strong impact.

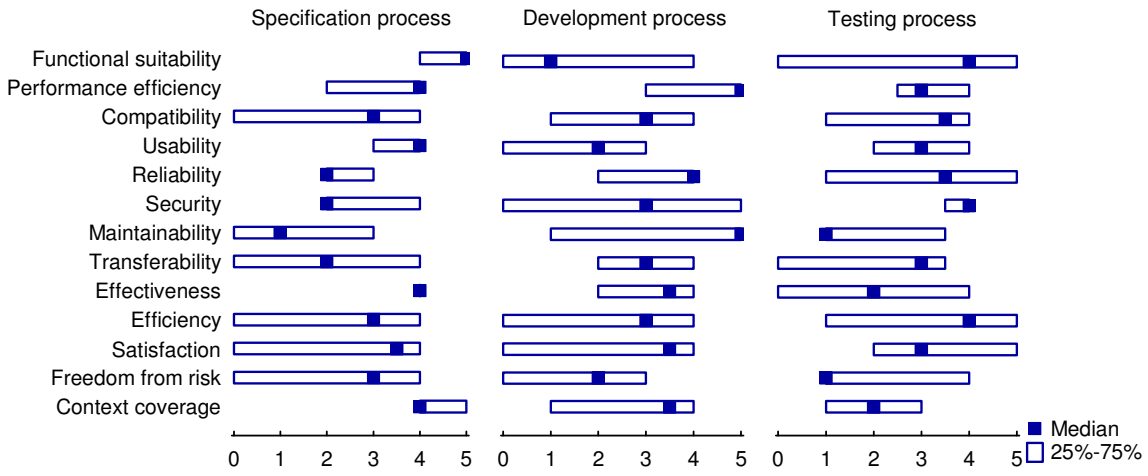

Figure 6. The impact of development process on quality features

4.5. Relationships among process factors

In this stage of questionnaire we investigated detailed relationships among process factors. This stage is the only one that did not include questions directly related to quality features but only those on the development process. Here, the questions have been divided into two levels. The aim at the higher level was to determine the importance of four main groups

of process-related factors on aggregated process quality, separately for three main development activities:

- Specification – covering any level involving specification;
- Development – covering any activities related to producing source code;
- Testing – covering various levels of software testing.

The four main groups of process-related factors are: *effort*, *process quality*, *people quality*, and *process difficulty*. At the lower level, each of these groups except effort has been decomposed into a set of detailed factors. This decomposition and the importance of factors at these two levels are illustrated in Figure 7. The left side contains four main groups of process related factors mentioned earlier. The right side contains detailed process factors grouped together with dashed lines pointing to a specific factor from a higher level.

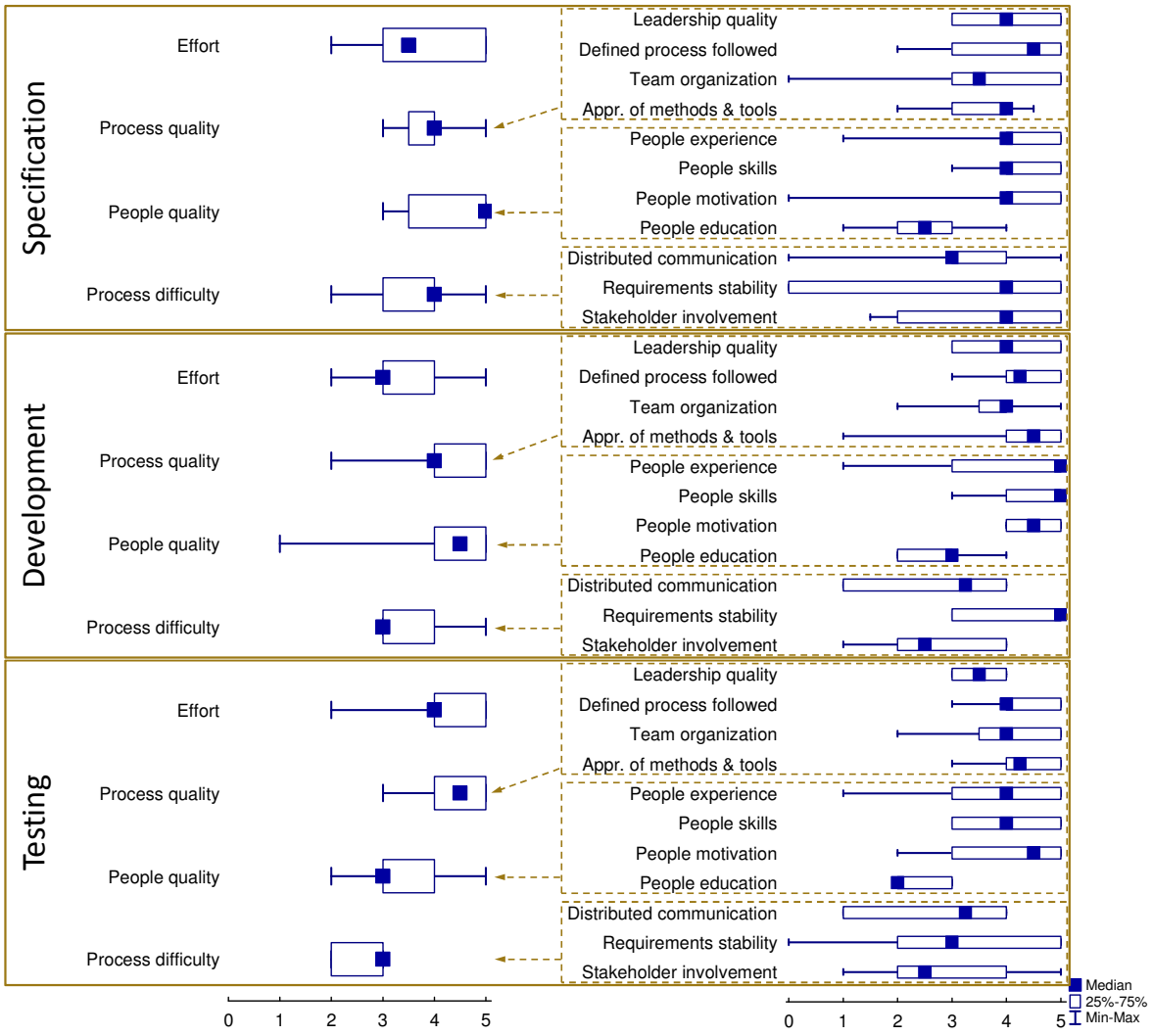

Figure 7. Detailed relationships between process factors

According to our respondents, *people quality* has the highest impact on *aggregated process quality* of specification activities. Other factors on average have similar strength of impact but they differ with the variability – *process quality* has the lowest variability.

For the development activities, *process quality* and *people quality* seem to have a higher impact on *aggregated process quality* than *effort* and *process difficulty*. For testing activi-

ties, the differences between factors are varying the most. *Process quality* and *effort* seem to be the most important, *people quality* with the moderate impact, and *process difficulty* with the least impact.

On the lower level (right side of Figure 7) we can also observe some differences in importance of factors depending on development activities. For example, the impact of detailed factors for *process difficulty* is varying strongly for specification activities and slightly less for testing. However, for development activities the impact of these factors seem to be more polarized – *requirements stability* significantly more important for *process difficulty* than *stakeholder involvement* and *distributed communication*.

4.6. Impact of project factors on quality features

The last part of questionnaire survey was focused on assessing the impact of project factors on quality features. Figure 8 illustrates a map of such impact where a darker color indicates stronger impact. In this case the strength of impact is expressed as a relative frequency at which respondents indicated particular project factor as influential on particular quality feature. The most commonly pointed relationship was between *deployment platform* and *performance efficiency* – all our respondents indicated this pair as in significant relationship. The following pairs were pointed by 88% of respondents: *architecture* → *performance efficiency*, *architecture* → *security*, *deployment platform* → *security*, *UI type* → *usability*, and *UI type* → *satisfaction*. There was just one pair of variables in this map, i.e. *UI type* → *freedom from risk*, that was not pointed by any respondent. 16 other pairs of variables in this map were pointed by just one respondent.

Project factor / Quality feature	Architecture	CASE tool usage	Deployment platform	UI type	Target market	Used methodology	Project difficulty
Functional suitability							
Performance efficiency							
Compatibility							
Usability							
Reliability							
Security							
Maintainability							
Transferability							
Effectiveness							
Efficiency							
Satisfaction							
Freedom from risk							
Context coverage							

Figure 8. Map of the impact of project factors on quality features (darker color indicates stronger impact)

Deployment platform and *architecture* on average are the most influential on all quality features, i.e. received the most counts column-wise. On the other hand, *CASE tool usage* and *target market* appear to be on average the least influential on quality features – they received the least counts column-wise. It can also be noticed that, according to respondents,

these project factors have much weaker impact on the quality-in-use features – the last five rows in Figure 8 are on average with much lighter shade than remaining top eight rows.

The questionnaire contained questions about the strength of impact of particular values of project factors on quality features. Respondents were able either to use predefined values, for example 'standalone', 'client-server' or 'multi-tier' for *architecture* or provide use their own values. Because several respondents chose to use their own classification for these project factors, obtained results cannot be easily aggregated. Such custom classifications were typically used by single respondents.

5. Lessons learned and threats to validity

One of the main problems in this study is the low number of respondents. Thus, it is difficult to use more advanced quantitative analytical techniques that require more cases at input. In addition, respondents represented diverse organizations, i.e. developing different types of software and using different processes. However, we also note that the primary goal of this survey was not to gather extensive knowledge on a wide range of projects and software organizations. Instead, we were looking to analyze the variability of answers provided by various experts.

Furthermore, the questionnaire contained questions aimed at calibrating a specific Bayesian network model. Thus, the answers provided by our respondents may be biased and would have been different for a model with another scope and structure.

The group of respondents was not homogeneous in terms of their background, age and experience. For example, some respondents consider themselves more as managers than software engineers. We also observed differences in respondents' attitude to this survey. Typically, those who were more strongly interested with potential usage of such model for predictions and simulations, were also more focused on providing carefully considered answers.

We informed respondents that, although the questionnaire has a structured form, respondents may put additional notes or provide explanations verbally. Some respondents used it to provide information on additional elements such as:

- Process activities – i.e. apart from specification, development and testing;
- Detailed process factors – typically related to process and people quality;
- Project factors – especially w the factors provided in the questionnaire were not relevant to the nature of projects developed in their organizations.

Usually respondents provided answers inconsistent with other respondents. The differences covered not only the strength of relationships between investigated factors but also the existence of relationship between certain pairs of factors. This exposed the general problem of acquiring subjective knowledge and aggregating the results. The sources of these differences partially can be explained by the fact that respondents participate in developing diverse software projects in different software organizations. Therefore, it may not be sensible to build a single simulation and predictive model but rather more tailored models for different types of projects or organizations.

6. Conclusions and future work

The analysis performed in this study lead to the following conclusions:

1. There is a demand in industry to develop predictive and simulation models covering a wider range of quality features. However, there are organizations and projects where such models are not relevant and useful.

2. When assessing strength of impact of particular factors on quality features, most experts used a predefined set of factors, although they were able to provide other factors according to their point of view. Together with the fact that these relationships were often seen as moderate, strong or very strong, this ensures us that the factors that will be used in the model have been selected appropriately.
3. There is a high variability in experts' perception on the factors that influence software quality. While some experts may consider a particular factor as important for a given quality feature, other experts may see such factors as not very important or even further – as influencing with an opposite direction. Thus, it may be very difficult to aggregate these results into a single simulation and predictive model.

Overall, we believe that results discussed in this paper and the model may provide useful support to decision makers in software projects. In future, we plan to extend this research by combining these results from survey among software experts with more objective empirical data, as well as to perform more detailed validation of the simulation model. Due to a high variability of responses we also plan to investigate the possibility of developing models tailored to individual needs and then to evaluate the usefulness of such models.

Acknowledgement

I would like to thank all participants of the questionnaire survey for providing useful information and additional thoughts inspiring this and further research. This work has been supported by research funds from the Ministry of Science and Higher Education and the National Science Centre in Poland as a research grant no. N N111 291738 for years 2010-2012.

References

[1] Abouelela M., Benedicenti L., *Bayesian Network Based XP Process Modelling*, International Journal of Software Engineering and Applications, vol 1, no.3, pp. 1-15, 2010.
[2] Beaver J.M., *A life cycle software quality model using Bayesian belief networks*, Doctoral Dissertation, University of Central Florida, Orlando, FL, 2006.
[3] Fenton N., Hearty P., Neil M., Radliński Ł., *Software Project and Quality Modelling Using Bayesian Networks*. In: Meziane, F., Vadera, S. (eds.) Artificial Intelligence Applications for Improved Software Engineering Development: New Prospects, Information Science Reference, New York, pp. 1-25, 2008.
[4] Fenton N., Neil M., Marsh W., Hearty P., Radliński Ł., Krause P., *On the effectiveness of early life cycle defect prediction with Bayesian Nets*, Empirical Software Engineering, vol. 13, pp. 499-537, 2008.
[5] ISO/IEC 25010:2011(E), *Software engineering – Software product Quality Requirements and Evaluation (SQuaRE) – System and software quality models*, 2011.
[6] Jones C., *Applied Software Measurement: Global Analysis of Productivity and Quality, Third Edition*, McGraw-Hill, New York, 2008.
[7] Kan S. H., *Metrics and Models in Software Quality Engineering*, Addison-Wesley, Boston, 2003.
[8] Lyu M., *Handbook of software reliability engineering*, McGraw-Hill, Hightstown, NJ, 1996.
[9] Maxwell, K.D.: *Applied Statistics for Software Managers*. Prentice Hall PTR, Upper Saddle River, 2002.
[10] Radliński Ł., *A conceptual Bayesian net model for integrated software quality prediction*, Annales UMCS, Informatica, vol. 11, no. 4, pp. 49-60, 2011.

[11] Radliński Ł., *A Framework for Integrated Software Quality Prediction using Bayesian Nets*, in Proceedings of International Conference on Computational Science and Its Applications (ICCSA 2011), Santander: Springer, 2011.
[12] Radliński Ł., *Empirical Analysis of the Impact of Requirements Engineering on Software Quality*, Requirements Engineering: Foundation for Software Quality, Lecture Notes in Computer Science, vol. 7195, Springer, Berlin-Heidelberg, pp. 232-238, 2012.
[13] Radliński Ł., *Enhancing Bayesian Network Model for Integrated Software Quality Prediction*, in Proc. Fourth International Conference on Information, Process, and Knowledge Management, Valencia, 2012, pp. 144-149.
[14] Radliński Ł., *Factors of Software Quality – Analysis of Extended ISBSG Dataset*, Foundations of Computing and Decision Studies, vol. 36, no. 3-4, pp. 293-313, 2011.
[15] Van Koten C., Gray A.R., *An application of Bayesian network for predicting object-oriented software maintainability*, Information and Software Technology, vol. 48, pp. 59-67, 2006.
[16] Wagner S., *A Bayesian network approach to assess and predict software quality using activity-based quality models*, In: 5th Int. Conf. on Predictor Models in Software Engineering, ACM Press, New York, 2009.
[17] Zhang D., Tsai J. J. P., *Machine Learning and Software Engineering*, Software Quality Journal, vol. 11, no. 2, pp. 87-119, 2003.