Software Risk Management in Practice: Shed Light on Your Software Product

Jens Knodel[1], Matthias Naab[1], Eric Bouwers[2], Joost Visser[23]

[1]Fraunhofer Institute for Experimental Software Engineering (IESE)
Kaiserslautern, Germany
{jens.knodel, matthias.naab}@iese.fraunhofer.de

[2]Software Improvement Group (SIG)
Amsterdam, The Netherlands
{e.bouwers, j.visser}@sig.eu

[3]Radboud University
Nijmegen, The Netherlands
j.visser@cs.ru.nl

Abstract—**You can't control what you can't measure. And you can't decide if you are wandering around in the dark. Risk management in practice requires shedding light on the internals of the software product in order to make informed decisions. Thus, in practice, risk management has to be based on information about artifacts (documentation, code, and executables) in order to detect (potentially) critical issues.**

This tutorial presents experiences from industrial cases world-wide on qualitative and quantitative measurement of software products. We present our lessons learned as well as consolidated experiences from practice and provide a classification scheme of applicable measurement techniques.

Participants of the tutorial receive an introduction to the techniques in theory and then apply them in practice in interactive exercises. This enables participants to learn how to shed light on the internals of their software and how to make risk management decisions efficiently and effectively.

I. Key Messages of the Tutorial

This full-day tutorial "Software Risk Management in Practice" is highly interactive with group exercises and discussions. It conveys the following key messages:

- **Evaluate your software product – early and regularly!**
- **Select appropriate methods and allocate effort based on goals and context**
- **Be aware what you measure and which criteria you use**
- **Be quantitative where possible, qualitative otherwise**
- **Use measurements to support decisions**
- **Carefully interpret results and put them into action**

This tutorial takes a practical perspective on risk management derived from experiences and lessons learned in various projects with industry.

- **Practitioners** learn how to apply theory on software product evaluation in practice.
- **Students** learn about the state of the practice and get data to guide them in their subsequent research.
- **Researchers** gain insights into the gap between state of the art and state of the practice and find new research challenges.

More information about the tutorial can be found at: www.sig.eu/Research/SoftwareRiskManagementInPractice

II. About the Tutorial

Format: At the start of the tutorial, we collect expectations and previous knowledge of the participants. The introductory part of the presentation with some theoretical foundations are aligned with the experiences and the background of the participants. The topic is introduced by discussing practical questions. Further examples are discussed and selected tasks are conducted together with the participants in interactive sessions. This also motivates the exchange of experiences among the participants.

Material: All participants receive an electronic copy of the slides in PDF format.

Intended audience: We aim at having 15-25 participants from different topic areas (e.g., software and system architecture, maintenance and evolution) from industry (e.g., system or service providers, service integrators, tool developers, service users) and academia (maintenance, evolution, or architecture research).

III. Structure of the Tutorial

- **Introduction**
  Introduction of speakers and participants and their respective background
- **Overall Context**
  Software product lifecycle (requirements - architecture - implementation – executable)
  Introduction of goals, motivation, and related terminology
- **Measuring Requirements**
  Concern Elicitation Check
- **Measuring Architecture**
  Solution Adequacy Assessment , Documentation Assessment, Compliance/Distance Assessment
- **Measuring Implementation**
  Maintainability model, Code quality assessment, models for other ISO 25010 aspects
- **Measuring Executable**s
  SPR models versus live measurements

- **Assessment Project Management**
  Initiation, set-up, outcome, follow-up
- **Wrap-up**
  How do you plan to apply the knowledge gained today?

IV. Dimensions of Software Product Evaluation

A software measurement technique defines measurement instructions to mitigate (detect) technical risks in a software (-intensive) product. It is applied on one evaluation object (or on several) and is measured against a baseline

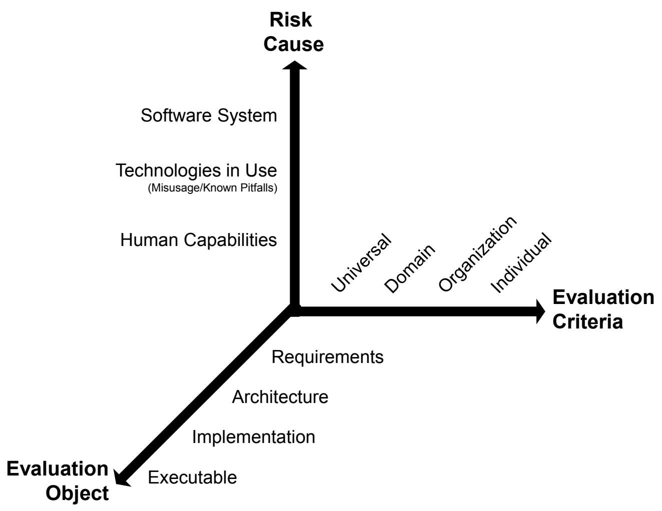
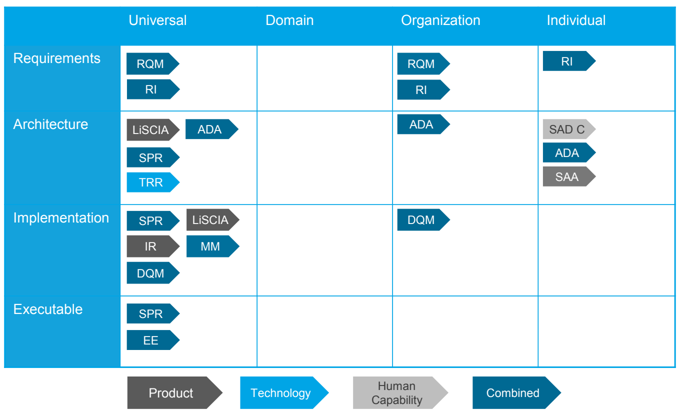


Figure 1: Dimensions of Software Product Evaluation

Software product evaluation methods can be categorized along at least three dimensions Figure 1.

- **Evaluation Object:** The most important dimensions is the object under evaluation. We can evaluate requirements, architecture, implementation, or the final executable software system.
- **Evaluation Criteria:** The criteria against which the object is evaluated can range from universal criteria (applicable in the same form to any object, e.g. ISO Standard) to individual criteria (tuned specifically to the object under evaluation, e.g. meeting product-specific performance requirements).
- **Risk Cause:** Risks that can be detected with evaluation can stem from different origins. Possible risk causes include the software system itself, technologies used in its construction, and limitations in the capabilities of humans dealing with the system (e.g. users, developers, maintainers, or operators).

Software product evaluation methods can be categorized in the space defined by these dimensions. Figure 2 shows a sample of product evaluation methods developed by Fraunhofer IESE and SIG. Table 1 explains the abbreviations used.



	Universal	Domain	Organization	Individual
Requirements	RQM, RI		RQM, RI	RI
Architecture	LiSCIA, ADA, SPR, TRR		ADA	SAD C, ADA, SAA
Implementation	SPR, LiSCIA, IR, MM, DQM		DQM	
Executable	SPR, EE			

Product
Technology
Human Capability
Combined

Figure 2: Example Software Product Evaluation Methods

Table 1: Abbreviations Explained

Term	Explanation
ADA	Architecture Documentation Assessment
DQM	Design Quality Metrics
EE	Energy Effeciency Model
IR	Issue Resolution Time
LiSCIA	Lightweight Sanity Check for Implemented Architectures
MM	Maintainability Model
RI	Requirements Inspection
RQM	Requirements Quality Metrics
SAA	Solution Adequacy Assessment
SAD C	Software Architecture Document Check
SPR	Security, Performance, Reliability model
TRR	Technical Risk Reviews

V. About the Presenters

Fraunhofer IESE and the Software Improvement Group (SIG) join their competencies for this tutorial. Previous tutorials of Fraunhofer IESE and SIG are combined and aligned to ensure a more comprehensive perspective on practical risk management.

A. Fraunhofer IESE

Dr. Jens Knodel and Dr. Matthias Naab are software architects. Their expertise – consolidated experiences and lessons learned from more than 50 projects with industry in domains like Embedded Systems, Information Systems, and Smart Ecosystems – lies in the definition, improvement, and assessment of software architectures.

Jens and Matthias are senior researchers at the Fraunhofer Institute for Experimental Software Engineering IESE in Kaiserslautern, Germany. They are responsible for project management, method development, and technology transfer in research and industry projects and are leading research activities in the area of software and systems architecture at IESE.

In addition, Jens Knodel, and Matthias Naab regularly coach practitioners on software architecture. They also give tutorials at conferences and hold lectures at the Fraunhofer Academy and at the University of Kaiserslautern on the same topics. They are the authors of more than 50 scientific, peer-reviewed publications in the areas of software architecture, maintenance, and evolution.

B. Prior Tutorials Given by the Fraunhofer IESE Presenters

- Conference "Software Engineering 2010", Paderborn, Germany
- Conference "CSMR 2011", Oldenburg, Germany
- Conference "Software Engineering 2013", Aachen, Germany (30 participants)
- Conference "Software Engineering 2014", Kiel, Germany (25 participants)
- Conference "WICSA 2014", Sydney, Australia (16 participants)
- "Seminar Software Architecture" – Fraunhofer Academy, Kaiserslautern, Germany, seminar held twice per year
- For many of our industrial customers on several occasions

C. Software Improvement Group

Eric Bouwers is a qualified teacher and technical consultant at the Software Improvement Group in Amsterdam, The Netherlands. He is interested in how software metrics can assist in quantifying the architectural aspects of software quality. In the past six years, this interest has led to the design, evaluation, and application of two architecture-level metrics that are now embedded in a benchmark-based model for software quality.

Joost Visser is head of research at the Software Improvement Group (SIG) in Amsterdam, The Netherlands, and holds a position as professor of large-scale software systems at Radboud University Nijmegen, The Netherlands. At the SIG, Joost is responsible for innovation of tools and services, academic relations, internship coordination, and general research. In the past eight years, he has been involved in the development, evaluation, and application of a benchmark-based model for software quality.

D. Prior Tutorials Given by the SIG Presenters

- Name: Software Measurement Pitfalls & Best Practices, Given at the 35th International Conference on Software Engineering (ICSE 2013),
- Integration of architecture measurement as part of the Software Architecture course of TU Delft in the past two years (see http://avandeursen.com/2013/12/30/teaching-software-architecture-with-github/).
- Tailored measurement workshops about software (architecture) measurement for different industrial customers.