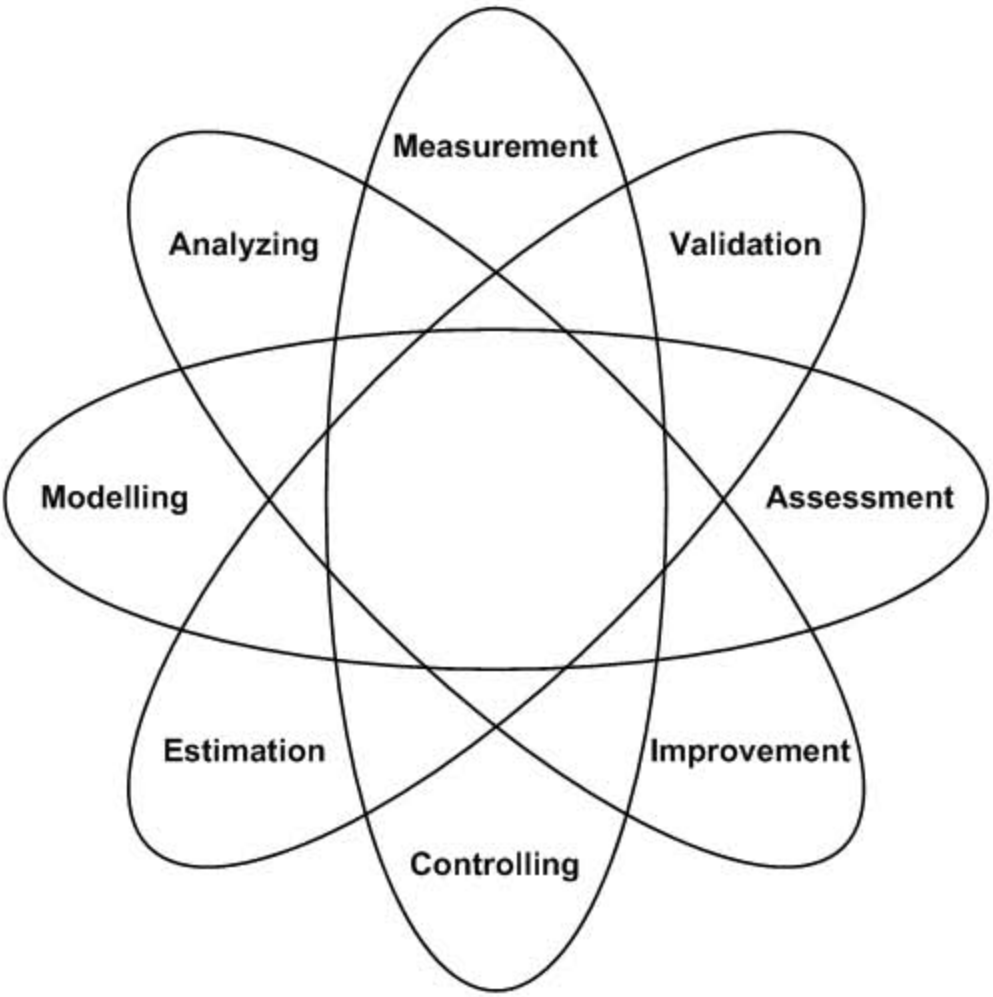
DASMA Volume 10, Number 2, December 2005

METRICS NEWS

Journal of the Software Metrics Community



Editors:

Alain Abran, Manfred Bundschuh, Reiner Dumke, Christof Ebert, Horst Zuse

OTTO-VON-GUERICKE-UNIVERSITÄT MAGDEBURG

Université du Québec
École de technologie supérieure

SM L@b

COSMIC-FFP
Common Software Measurement International Consortium

The *METRICS NEWS* can be ordered directly from the Editorial Office (address can be found below).

Editors:

Alain Abran
Professor and Director of the Research Lab. in Software Engineering Management
École de Technologie Supérieure - ETS
1100 Notre-Dame Quest,
Montréal, Quebec, H3C 1K3, Canada
***Tel.:** +1-514-396-8632, **Fax:** +1-514-396-8684*
aabran@ele.etsmtl.ca

Manfred Bundschuh
Chair of the DASMA
Sander Höhe 5, 51465 Bergisch Gladbach, Germany
***Tel.:** +49-2202-35719*
bundschuhm@dasma.de
http://www.dasma.org

Reiner Dumke
Professor on Software Engineering
University of Magdeburg, FIN/IVS
Postfach 4120, D-39016 Magdeburg, Germany
***Tel.:** +49-391-67-18664, **Fax:** +49-391-67-12810*
dumke@ivs.cs.uni-magdeburg.de

Christof Ebert
Dr.-Ing. in Computer Science and Director SW Coordination
Alcatel HQ
54, Rue La Boetie, F-75008 Paris, France
***Tel.:** +33-675-091999, **Fax:** +33-1-4076-1475*
christof.ebert@alcatel.org

Horst Zuse
Dr.-Ing. habil. in Computer Science
Technical University of Berlin, FR 5-3,
Franklinstr. 28/29, D-10587 Berlin, Germany
***Tel.:** +49-30-314-73439, **Fax:** +49-30-314-21103*
zuse@tubvm.cs.tu-berlin.de

Editorial Office: Otto-von-Guericke-University of Magdeburg, FIN/IVS, Postfach 4120, 39016 Magdeburg, Germany

Technical Editor: DI Martin Kunz

The journal is published in one volume per year consisting of two numbers.

CALL FOR PAPERS

Université du Québec
École de technologie supérieure

IWSM2006

16th International Workshop
on Software Measurement

MetriKon2006

DASMA Metrik Kongress

November 2-3, 2006, Potsdam, Germany

THEME & SCOPE:

Software measurement and metrics are key technologies to manage and to control software development projects. Measurement is essential for any engineering activity, by increasing the scientific and technical knowledge for both the practice of software development and for empirical research in software technology. This congress facilitates the exchange of software measurement experiences between theory and practice.

TOPICS OF INTEREST:

We encourage submissions in any field of software measurement, including, but not limited to:

Software metrics foundations
Practical measurement application
Measurement processes and resources
Empirical case studies
Measurement acceptance
Functional size measurement
Software estimation
Software process improvement
Metrics for specific areas, e.g. for web services
Metrics for system engineering, integration, and testing
Measurement data bases
Metrics validation
Measurement services
Measurement tools
Measurement experience and guidance
Theory of measurement
Measurement paradigms
Enterprise embedded solutions
Software benchmarking

SUBMISSIONS:

Authors should send proposed *short papers (2 to 4 pages)* via eMail by June 19st, 2006 to

Alain Abran	***Günter Büren***	***Reiner Dumke***
ÉTS, Montréal, Canada	Büren & Partner, Nuremberg	University of Magdeburg, Germany
aabran@ele.etsmtl.ca	gb@bup-nbg.de	dumke@ivs.cs.uni-magdeburg.de

Papers should not have already been published elsewhere. Nor should they have been submitted to a journal or to another conference. At least one among the authors of each paper accepted should register for the conference and ensure paper presentation. Conference languages are English and German. German papers will be presented in a separate track.

CONFERENCE TIMETABLE:

Submission deadline of paper:	June 19, 2006
Notification of acceptance:	July 31, 2006
Final paper deadline:	September 11, 2006
Conference date:	November 1-2, 2006

CONTACT:

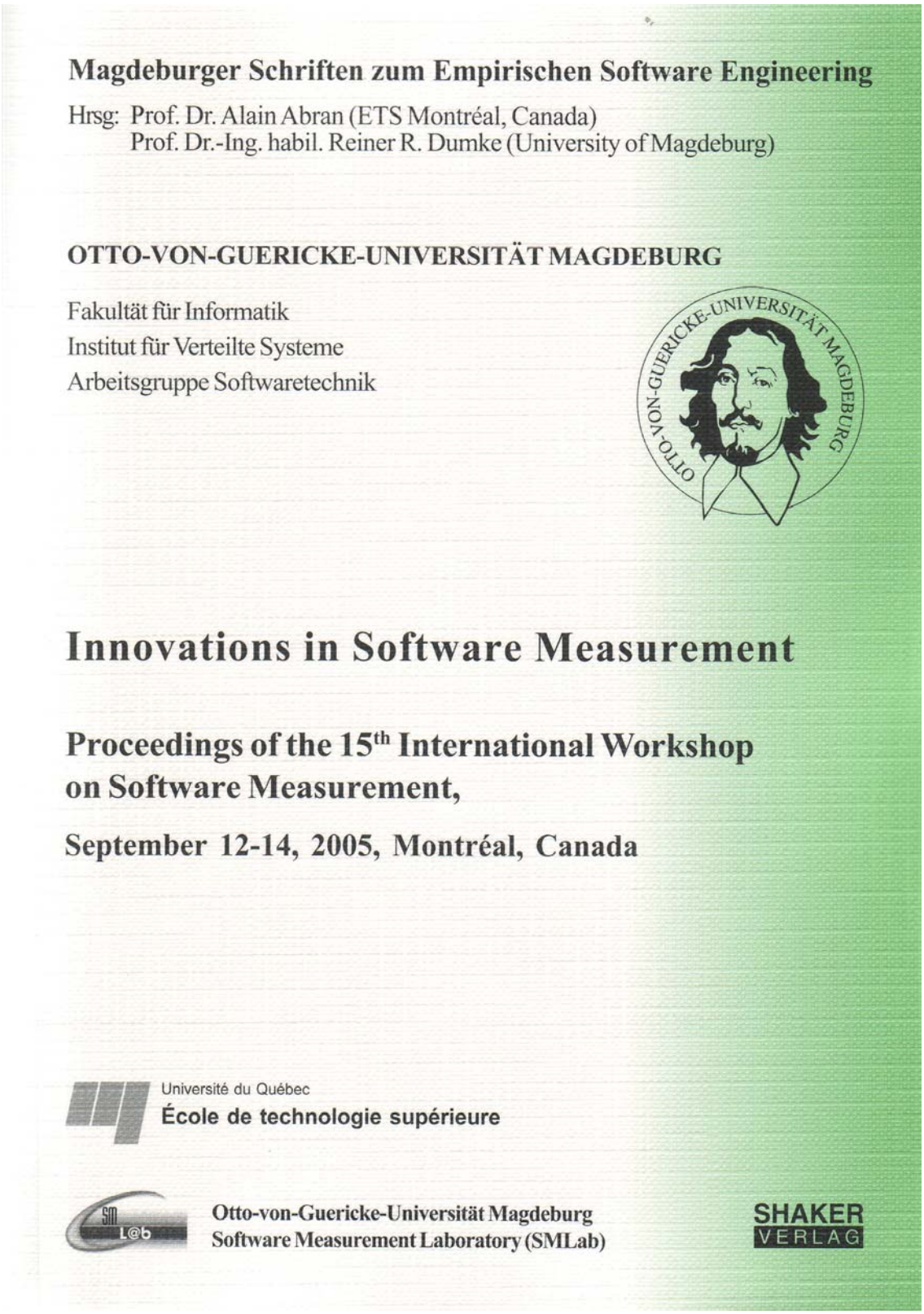
DASMA e.V.:
Romy Robra (Tel. +49 - 9126 - 29 79 576, info@dasma.org);

Uni Magdeburg:
Dagmar Dörge (Tel. +49 - 391 - 67 18 664, doerge@ivs.cs.uni-magdeburg.de)

FURTHER INFORMATION: (including author guidelines and templates)

http://www.dasma.org
http://iwsm2006.cs.uni-magdeburg.de
http://www.metrikon.de

Our 15th Workshop on Software Measurement (IWSM 2005) took place in Montréal, Kanada in September 2005. The following report gives an overview about the presented papers. Furthermore, the papers are published in the following Shaker book (ISBN 3-8322-4405-0):



Magdeburger Schriften zum Empirischen Software Engineering

Hrsg: Prof. Dr. Alain Abran (ETS Montréal, Canada)
Prof. Dr.-Ing. habil. Reiner R. Dumke (University of Magdeburg)

OTTO-VON-GUERICKE-UNIVERSITÄT MAGDEBURG

Fakultät für Informatik
Institut für Verteilte Systeme
Arbeitsgruppe Softwaretechnik

Innovations in Software Measurement

Proceedings of the 15th International Workshop on Software Measurement,

September 12-14, 2005, Montréal, Canada

Université du Québec
École de technologie supérieure

Otto-von-Guericke-Universität Magdeburg
Software Measurement Laboratory (SMLab)

SHAKER VERLAG

Towards Meaningful Metrics Data Bases

René Braungarten, Martin Kunz, Ayaz Farooq, Reiner R. Dumke

Software Engineering Group
Institute for Distributed Systems, University of Magdeburg,
P.O. Box 4120, 39106 Magdeburg, Germany

{braungar, makunz, farooq, dumke}@ivs.cs.uni-magdeburg.de
http://www.smlab.de/

Abstract. The importance of measuring artifacts emerging during the software development process is beyond controversy not only for economic purposes, these days. To expand those data towards empirical series of measurement and thus to benefit from it in the long-run, a structured and persistent acquisition is almost compulsory. Renowned proposals for measurement programs and models to assess an organization's measurement capability describe the prominence of measurements and suppose the usage of appropriate metrics data bases in different fields of application. But, analogous to common project management or process frameworks they leave organizations alone in providing guidelines or standards for the collection process, too. So we propose a Metrics Data Base Maturity Model (MDB^{MM}) bearing the potential to appraise an organization's measurement collection process' maturity and to expose leverage points for its improvement.
Withal, the conceived MDB^{MM} has its seeds in a field study concerning prevalent metrics data bases in practice to characterize their status quo as well as commonly applied infrastructures, on the one hand. On the other hand, it strongly avails itself of an investigation in factors influencing and MDB's maturity thereby not abstracting away from accepted frameworks like CMMI, V-Modell® XT, or Measurement-CMM. As a result, the MDB^{MM} enables us to appraise measurement collection process maturity with the aid of a maturity scale similar to that of CMMI-SE/SW simultaneously suggesting ways for improvement.

Architecture Maturity and Requirements Maturity Do not Explain Each Other

Maya Daneva

University of Twente

m.daneva@utwente.nl

Abstract. Enterprise Architecture and Enterprise Resource Planning (ERP) solutions have profound influences on today's connected business world. Yet, the relationship between the two is not enough explored and, as a result, only partially understood. This paper sheds some light into this relationship and suggests exploring it by using the concept of maturity. We build upon our previous experiences in assessing ERP Requirements Engineering processes and enterprise architecture processes in an industrial setting.

Tool supported effort monitoring and estimations in EAI multi projects

Daniel Reitz[#*], *Andreas Schmietendorf*[#*], *Reiner Dumke*[*]

[#]T-Systems International GmbH, SSC P&PA,
Wittestraße 30H, 13509 Berlin
{daniel.reitz, andreas.schmietendorf}@t-systems.com

[*]Otto-von-Guericke-Universität Magdeburg, Fakultät Informatik,
Institut für Verteilte Systeme, Postfach 41 20, 39016 Magdeburg,
{reitz, schmiete, dumke}@ivs.cs.uni-magdeburg.de

Abstract. This article is about effort monitoring and estimation in the context of a complex EAI solution. Therefore we want to introduce a multi project reporting environment to monitor the progress of different parts (sub-projects) of an EAI project. This environment is also used to store and present empirical data from realized EAI sub-projects to support the step by step development of a methodical estimation process. Furthermore we want to discuss within this document the problems and challenges of effort and cost estimations in EAI projects and we want to present a method to handle integration requirements in large enterprises with EAI activities.

KEYNOTE:

Offshoring – 6 years of industrial experience in distributed software development

Andreas Schmietendorf[#*], *Stanimir Stojanov*[+]

[#] T-Systems International, IBU Telco, SSC P&PA,
Wittestraße 30G, 13476 Berlin
andreas.schmietendorf@t-systems.com

[*] Hochschule Harz, University of applied science,
Friedrichstraße 57-59, 38855 Wernigerode
aschmietendorf@hs-harz.de

[+] University of Plovdiv, Faculty of Mathematics and Informatics,
24 Tzar Assen St, 4000 Plovdiv, Bulgaria
csstani@pu.acad.bg

Abstract. The so called "offshore development" is a new trend for the software development community. The aim is much of the development work is done in lower paid countries. Within this paper we describe selected aspects of a long-time partnership between an accomplishment unit of T-Systems and a Bulgarian cooperation partner. Since 1998, the authors implemented a procedure for distributed software development step-by-step. This form of cooperation in the area of the software development is connected with the concept of the offshoring today. More than 25 industrial projects could be realized successfully through this partnership.

Measurement eLearning
A classification approach for eLearning-Systems

Martin Kunz, René Braungarten, Reiner R. Dumke

Software Engineering Group
Institute for Distributed Systems
University of Magdeburg, Germany

{makunz, braungar, dumke}@ivs.cs.uni-magdeburg.de
http://ivs.cs.uni-magdeburg.de/sw-eng/us/

Abstract. With increasing competition in the software development industry the guidance of management decision with software measurement methods gets more and more important.
Due to the general complexity of these methods and the rising geographical distance of the participants of the software development process, web-based system for professional training and collaboration in the software engineering sector gains more relevance.
This paper describes a potential way of evaluation of such systems, and classifies a selected example with the presented three categories.

Software Maintenance expert system (*SM*^{xpert})
Measuring the use of the knowledge base

Alain April, Jean-Marc Desharnais

École de Technologie Supérieure, 1100 Notre-Dame West, Montreal, Canada

alain.april@etsmtl.ca, jean-marc.desharnais@etsmtl.ca

Abstract. Maintaining and supporting the software of an organization is not an easy task, and software maintainers do not currently have access to tools to evaluate strategies for improving the specific activities of software maintenance. This article presents a knowledge-based system which helps in locating best practices in a software maintenance capability maturity model (S3^{m}). It presents an XML-based usage of the knowledge base to measure the concepts most often employed by software maintainers. The contributions of this paper are: 1) to instrument the maturity model with a support tool to aid software maintenance practitioners in locating specific best practices; and 2) to describe an XML-based measurement approach to locate the concepts most often accessed by users.

On the Applicability of FPA and COCOMO II in a workflow and maintenance context

Valérie Paulus[1], Grégory Seront[1], Miguel Lopez[1], Christian Huvelle[2], Bernard Bolle[2]

[1]CETIC asbl, Rue Clément Ader, 8, B-6041 Gosselies, Belgium
vp, malm, gs@cetic.be

[2]Siemens Business Services n.v./s.a., Chaussée de Liège 221, 5100 Namur
christian.huvelle, bernard.bolle@siemens.com

Introduction

The estimation of the needed effort to develop a brand new application or a brand new series of functionalities is a day-to-day challenge in software development organisations. Some techniques exist and are used in the industry. Well known methods are the function point analysis (FPA) and the predictive cost model COCOMO II. Both are used in the development team of Siemens Business Services Belgium (SBS).

Benchmarking is an essential control mechanism for management

Ton Dekkers

Sogeti Nederland B.V.

ton.dekkers@sogeti.nl

Abstract. A great number of big organisations already has been outsourcing the IT activities or is thinking about outsourcing. Even outsourcing companies in some cases use specific variances in outsourcing IT, e.g. near-shore or offshore. But are these activities so beneficial, are they introducing other risks or is it all opportunity?
Management needs to find a way to decide on transparent and objective criteria whether outsourcing could be beneficial. First of all a measurement model for controlling outsourcing or investigating the cost benefit ratio for outsourcing should be adopted. Key issue in most of these models is project delivery rate. To be able to compare and to judge project delivery rates, benchmarking is a good way to resolve this topic.

Investigation of the Effort Data Consistency in the ISBSG Repository

David Dery, Alain Abran

École de Technologie Supérieure

david.dery.1@ens.etsmtl.ca, aabran@ele.etsmtl.ca

Abstract. To develop adequate software project estimation models using statistical techniques, the consistency of historical data is important. This paper investigates this issue by looking into the consistency of the information contained in one of the most important fields in the International Software Benchmarking Standards Group (ISBSG) repository, that is, the project effort data field. This paper also presents an example of how effort data from projects that include a large number of project phases can be used for extrapolation, through a normalization process, to projects with fewer phases. The ISBSG organization has attempted to tackle this issue on the variability of phases included in the project effort field by deriving a normalized work effort field. This paper investigates this problem and reports on a number of related issues.

MTPF Function Points Measure Early Method

Ramón Asensio Monge, Francisco Sanchis Marco, Fernando Torre Cervigón

Abstract. The function point analysis (FPA) by A.J. Albrecht is a standard method for development from the customer's point of view. International Function Point Users Group, (IFPUG), considers FPA as a standard in the software functional size measurement. IFPUG follows Albrecht's method and adds in its releases modifications to its rules and hints in order to improve it. The documentation level required for the implementation of this method corresponds to Rudolph's classification level I (in requirement specifications). This documentation is hardly available to development software organizations for others when budgeting. It represents a problem as well for those companies developing their own software if they are forced to quit a project development due to requirement specification process costs. The present work aims to develop an early method of function point measure of software products functional size, whose acronym is MTPF. The required documentation to implement MTPF is some documentation obtained previous the Analysis activity. The results of the implementation of MTPF method will be presented in this article, divided in our leading point:

- Determination of the factors to be considered for MTPF method implementation
- MTPF method implementation
- MTPF measure and supporting tools
- Determination of MPTF method goodness

A Case Study on Using Functional Size Measurement Methods for Real Time Systems

Cigdem Gencel, Onur Demirors, Erhan Yuceer

cgencel@ii.metu.edu.tr, demirors@ii.metu.edu.tr, Erhany@AYESAS.com

Abstract. Among various approaches to software size estimation, the metrics and methods based on "functionality" have been widely used. Although all Functional Size Measurement (FSM) methods measure size in terms of the "functionality" provided to the users, what they count and how they do change according to the type of application domain the method is designed to be applicable, software entity types used to measure the size attribute and the phase of estimation during the software development life cycle. This paper presents the results obtained by applying two well-known FSM methods; Mk II FPA 1.3.1 and COSMIC FFP 2.2 to a real-time system which have control as well as algorithmic components. These methods are selected not only for being international ISO standards that are conformant to ISO/IEC 14143, but have measurement manuals as well. The comparison of these methods with respect to their measurement processes, the difficulties faced during the measurement processes and the improvement opportunities for FSM approaches are also discussed.

FSM Influences and Requirements in CMMI-Based Software Processes

Reiner R. Dumke, Karsten Richter, Thomas Fetcke

University of Magdeburg, Institute of Distributed Systems, Magdeburg, Germany

dumke@ivs.cs.uni-magdeburg.de
http://ivs.cs.uni-magdeburg.de/sw-eng/agruppe/

Abstract. The following paper describes the influences of functional size measurement (FSM) methods in the Capability Maturity Model Integration (CMMI). Furthermore, it is indicated where FSM is necessary in CMMI-based process improvement. Starting with a short characterization of the functional size measurement approach we discuss the role and intention of FSM during the software processes.
Based on an industrial project using CMMI evaluation we will present an analysis of the parts and ingredients of functional size considering the different process maturity levels.

Adapting Use Case Model for COSMIC-FFP Based Measurement

Piotr Habela[1], Edgar Głowacki[1], Tomasz Serafiński[2], Kazimierz Subieta[1 2 3]

[1] Polish-Japanese Institute of Information Technology, Warsaw, Poland
{piotr.habela, edgar.glowacki}@pjwstk.edu.pl

[2] Łódź University of Technology,
Łódź, Poland,
tomaszek@kis.p.lodz.pl

[3] Institute of Computer Science, Polish Academy of Sciences;
Polish-Japanese Institute of Information Technology, Warsaw, Poland
subieta@ipipan.waw.pl

Abstract. The COSMIC-FFP estimation is defined using abstract, domain-independent terms that need to be mapped onto the notions of the utilized software development methodology. On the other hand, for productivity reasons it is important to avoid model reconstruction dedicated for just measuring purposes. Use Case model seems to be an optimum candidate for serving both general purpose requirement management and measurement input model needs. The measurement preparation influences to some extent the perspective of the use case model, but can also enforce its quality. In this paper we describe our experiences in adjusting both use case model style and detailed measurement principles to achieve synergy between requirement specification and size estimation, and to keep the counting intuitive and adequate. We focus on resolving the functional redundancy issue by appropriate Use Case organization. In addition, the suggested style of use case specification is intended to make allow counting the functional size units in a straightforward way for particular scenario step specifications. Some remarks on counting use case variants and different actor sets are also provided.

COSMIC-FFP & Functional Complexity (FC) Measures: A Study of their Scales, Units and Scale Types

*Manar Abu Talib**, Alain Abran*, Olga Ormandjieva***

*École de Technologie Supérieure - ETS
1100 Notre-Dame Ouest, Montréal, Canada H3C 1K3
alain.abran@ele.etsmtl.ca

**Concordia University
1455 de Maisonneuve Blvd. W. Montreal, Quebec H3G 1M8, Canada
m_abutal@cse.concordial.ca, ormandj@cse.concordia.ca

Abstract. This paper presents an overview of some measurement concepts across both COSMIC-FFP, an ISO standard (ISO/IEC 19761) for functional size measurement and Functional Complexity (FC), an entropy-based measure. It investigates in particular three metrological properties (scale, unit and scale type) in both of these measurement methods.

Measurement Convertibility - From Function Points to COSMIC-FFP

Alain Abran, Jean-Marc Desharnais, Fatima Aziz

École de Technologie Supérieure - ETS
1100 Notre-Dame Ouest, Montréal, Canada H3C 1K3

alain.abran@etsmtl.ca, jean-marc.desharnais@etsmtl.ca, fatima.aziz.1@ens.etsmtl.ca

Abstract. Several organizations are interested in using convertibility ratios between COSMIC- FFP (ISO 19761), the second generation of functional size of the software, and Function Points Analysis – FPA (ISO 20926). This paper presents a survey of previous convertibility studies and reports on findings from an additional data set. In summary, these studies indicate that convertibility can be simple and straightforward when only human users are taken into account in the measurement viewpoint. It also provides indication that convertibility can be less straightforward is some instances.

Improvement of analysis model by removing improper parts based on functional size measurement

Shin-ichi Nagano[†]*, Tuneo Ajisaka*[‡]

[†]NTT 9-11 Midori-cho 3-chome Musashino-shi, Tokyo, 180-8585 Japan
nagano.s@lab.ntt.co.jp

[‡]Faculty of Systems Engineering,Wakayama University 930 sakaedani, wakayama-shi, Wakayama, 640-8510 Japan
ajisaka@sys.wakayama-u.ac.jp

Abstract. We propose a method to remove possible improper parts from an analysis model based on measured functional size tendencies of each function and object that constitute the model. Actual review shows that frequently occurring troubles include ambiguity, defectiveness, inconsistency, and so on. In light of this, we propose a method for removing possible inappropriate elements from an analysis model. The proposed method uses the functional size measurement method. This paper shows inadequacies in analysis models based on actual reviewing research and explains a method for removing possible improper parts. Additionally, the improvement of the analysis model will be discussed concretely and illustrated with simple examples.

Functional details visualization and classification in the COSMIC FSM framework

Luca Santillo

Data Processing Organization

luca.santillo@dpo.it

Abstract. Given the relevance of software functional size measurement to the industry and in practice, improved ways to represent and handle the measurement details can provide significant advantages and make possible to discover new interpretation and exploitation possibilities over measurement results. This work illustrates some enhancement proposal in the COSMIC framework, to improve the significance of the measures and to add such exploitation possibilities, by means of a so-called "data movement diagram" and a further extension, to include data manipulation classification for functional processes within the software, denoted "functional processing diagram". Possible advantages of the proposal would be to allow process-to-process comparison and profiling, complexity and reuse evaluation, and visual impact analysis of software changes required by enhancement projects.

Complex Evaluation of an Industrial Software Development Project

Andreas Schmietendorf[#*], *Reiner Dumke*[*]

[#]T-Systems International, Entwicklungszentrum Berlin, Integration Services,
Wittestraße 30G, 13476 Berlin
andreas.schmietendorf@t-systems.com

[*]Otto-von-Guericke-University Magdeburg, Fakultät Informatik,
Institut für Verteilte Systeme, Postfach 41 20, 39016 Magdeburg
{schmiete, dumke}@ivs.cs.uni-magdeburg.de

Abstract. Benchmarks are widely used to verify the maturity of project organizations. This paper shows our experiences with the implementation of a project related assessment. The assessment was driven from the wish to receive more transparency within an introduced project organization. We used as method for the evaluation our own benchmark process, first introduced on the IWSM 2003. During the last two years we applied this method in 6 industrial projects. This benchmark based on the identification of the process maturity, the realization of a strengths and weaknesses profile and the size measurement of the whole implementation. Based on the size measurement we derived the project related effort by the use of the COCOMO and Function Points method. Finally we compare the effort estimation with the real effort.

Analysis of Object-Oriented Metrics

K.K. Aggarwal, Yogesh Singh, Arvinder Kaur, Ruchika Malhotra

School of Information Technology, GGS Indraprastha University,
Delhi 110006, India

Abstract. The increasing importance of software measurement has lead to development of new software measures. Many metrics have been proposed related to various object-oriented constructs like class, coupling, cohesion, inheritance, information hiding and polymorphism. But there is a little understanding of the empirical hypotheses and application of many of these measures. It is often difficult to determine which metric is more useful in which area. As a consequence, it is very difficult for project managers and practitioners to select measures for object-oriented systems. In this paper we investigate 22 metrics proposed by various researchers. The metrics are first defined and then explained using practical applications. They are applied on sample projects on the basis of which descriptive statistics of each measure is presented. Finally, a review of the empirical study concerning chosen metrics and subset of these measures that provide sufficient information is given and metrics providing overlapping information are excluded from the set.

Measurement of Cohesion and Coupling in OO Analysis Model Based on Crosscutting Concerns

O. Ormandjieva, M. Kassab, C. Constantinides

Concordia University
1455 de Maisonneuve Blvd. W. Montreal, Quebec H3G 1M8, Canada

{ormandj,moh_kass,cc}@cse.concordia.ca

Abstract. Separation of Concerns is a fundamental software engineering principle achievable through implementation of the software development quality patterns Low Coupling and High Cohesion throughout the software development life-cycle. This paper introduces measurements for controlling the coupling and cohesion of the Object-Oriented (OO) analysis model based on the notion of crosscutting concerns. Controlling the OO analysis model's quality is crucial as errors introduced in the OO analysis model might propagate throughout the software development phases into the final product where their correction would require considerable additional effort and resources. The measurement control mechanisms for obtaining early feedback on the levels of coupling and cohesion in the analysis model help identify early crosscutting implications in the system. Associating the analysis model with these measurements leads to early feedback on the realization of the crosscuttings within the proposed system and thus an early possible treatment. The proposed cohesion measurement is new; the coupling measurement is an adoption of an existing OO design measure.

Information Management for Industrial eLearning Projects

Uwe Blazey[1], Reiner Dumke[2]

[1]UBISNET (Germany)
uwe@blazey.de

[2]Otto-von-Guericke-Universität Magdeburg
Institut für Verteilte Systeme
dumke@ivs.cs.uni-magdeburg.de

Abstract. To ensure a successfully management of eLearning projects, it is necessary to get current information of completed, running and future project- activities in well-structured quality. In the following there will be shown a view of the first activities with this targets. All used data and information are based on the UBISNET eLearning project database. The goal is, in view of the permanent increasing of complexity, to minimize the future project risks, to optimize the processes, hold the quality level and make improvements possible, where improvements are necessary. These works can be only successfully, if here relevant and measurable results are computed, on whose basis a correcting action in complex processes is possible. The beginnings introduced here, are part of a continuously quality and optimization management of UBISNET. The beginnings introduced here, are part of a continuously quality and optimization management of UBISNET and stand in context with subjects of software measurement and software assessment of the University of Magdeburg.

An Analysis of the Design and Definitions of Halstead's Metrics

Rafa E. Al Qutaish, Alain Abran

École de Technologie Supérieure, University of Québec,
1100 Notre-Dame Ouest, Montréal, Québec H3W 1T8, Canada

rafa.al-qutaish.1@ens.etsmtl.ca, alain.abran@etsmtl.ca

Abstract. Some software measures are still not widely used in industry, despite the fact that they were defined many years ago, and some additional insights might be gained by revisiting them today with the benefit of recent lessons learned about how to analyze their design. In this paper, we analyze the design and definitions of Halstead's metrics, the set of which is commonly referred to as 'software science'. This analysis is based on a measurement analysis framework defined to structure, compare, analyze and provide an understanding of the various measurement approaches presented in the software engineering measurement literature.

KEYNOTE:

Software Measurement Body of Knowledge – Overview of Empirical Support

Luigi Buglione, Alain Abran

École de Technologie Supérieure - ETS
1100 Notre-Dame Ouest, Montréal, Canada H3C 1K3

Luigi.Buglione@computer.org, alain.abran@ele.etsmtl.ca

Abstract. The "Guide to the Software Engineering Body of Knowledge – SWEBOK" (2004 version) – contains ten distinct Knowledge Areas (KAs) and three common themes: Quality, Tools and Measurement. Since measurement is present in all the KAs, an initial taxonomy for measurement had been proposed as a foundation for the addition of a new specific KA on Software Measurement. To verify the feasibility of such a proposal, this paper presents an overview of the level of empirical support for each measurement topic identified. The types of empirical support adopted are from the Zelkowitz & Wallace taxonomy.

Using Simulation to Determine the Sensibility of Error Sources for Software Effort Estimation Models

Mercedes Ruiz[1], Juan-José Cuadrado Gallego[2], Miguel-Angel Sicilia[2], Daniel Rodríguez[3]

[1]The University of Cádiz, Spain
mercedes.ruiz@uca.es

[2]The University of Alcalá, Spain
jjcg@uah.es, msicilia@uah.es

[3]The University of Reading, Reading, RG6 6AY, United Kingdom
d.rodriguezgarcia@reading.ac.u

Abstract. In this paper, a software project simulator based on System Dynamics is used to analyze the different sources of error in the initial estimates of parametric models. More precisely, the effect of using the two different COCOMO models, different calibration models, and a different selection of ratings for the effort multipliers in estimates is shown. The bdel-Hamid' system dynamics simulation model is used as a technique to observe the potential evolution of each error source during the lifecycle and its effects on the key variables of the project such as effort, productivity, schedule, etc. The study determines which of the three sources introduces a higher error in the initial estimations and also their effects in a project the end of the project. We observed how simulation helped to minimize the effect of these errors, corroborating the need for multiple estimation methods.

Independent Dimensions of Software Complexity

Roland Neumann[1], Dennis Klemann[2]

[1]Technical University Kaiserslautern, Software Engineering:
Dependability Group p.o.box 3049, 67653, Kaiserslautern, Germany
roland.neumann@informatik.uni-kl.de

[2]Hasso-Plattner-Institute for Software Systems Engineering GmbH at the
University Potsdam, 14482 Potsdam, Germany
dennis@klemann.org

Abstract. Assessing complex interactions and computation structures of a current software system needs the usage of metrics. Each metric gives information about an internal software property. However, each metric also influences other metrics due to their high correlation. One key problem is to determine this cross influence between metrics. For instance, the count of public methods in a class not only gives a size information but also information about possible interaction with other classes. What about assessing coupling issues independent from the length of methods or the count of attributes?
This paper describes how to work with independent metrics for assessing software complexity. Independent metrics can be used for gaining experience and comparing systems as described. It introduces a set of most characteristic software properties (complexity dimensions) drawn from empirical data, describes the techniques needed to compare metrics from different complexity surveys and shows practical application results to deepen the system's understanding.

The Measurement Service in Software Engineering Environments

Manuel Bollaín, Juan Garbajosa

Technical University of Madrid
System and Software Technology Group

mbollain@eui.upm.es, jgs@eui.upm.es

Abstract. It is commonly recognized that software engineering practice, to achieve a good performance, depends on measurement. While the automation of a number of software engineering activities are receiving a lot of attention, it is not the case for measurement. However a good automated support seems to be essential to consolidate the measurement process practice. At present a common understanding is that automation requires software engineering environments and properly integrated tools. This paper provides a first approach to provide a set of requirements for a software engineering environment measurement service, taking as a basis the on-going ISO work on software engineering environments services and INCOSE measurement tools survey criteria.

On the Impact of the Types Conversion in Java onto the Coupling Measurement

Miguel Lopez[1], Alain Abran[3], Grégory Seront[1], Naji Habra[2]

[1]CETIC asbl, Rue Clément Ader, 8, B-6041 Gosselies, Belgium
malm@cetic.be, gs@cetic.be

[2]Faculty of Computer Science, Namur University – FUNDP, Namur, Belgium
nha@info.fundp.ac.be

[3] Ecole de Technologie Supérieure, Montréal, Québec; Canada
aabran@ele.etsmtl.ca

Abstract. Software measurement represents an important topic heavily discussed within the software engineering community. Since thirty years, software measurement has become an important domain where interesting debates have occurred.
Internal measurements of software do not necessitate any execution. Since these measurements are automated, it is commonly accepted that during such measurements errors cannot occur. Indeed, such measurements have no random or probabilistic aspect.
The current paper aims at showing that other sources of error or uncertainty exist in the software measurement. Sources of uncertainty can appear before the measurement itself, that is, at the measurement design level. Indeed, mistakes related to the design of measurement can occur, and therefore affect the measurement results when executing the measures.
The current paper extends the notion of uncertainty to the measurement design level, and highlights the impact of the design uncertainty onto the measurement result.

An Agent-based Measurement Infrastructure

Reiner Dumke, Martin Kunz, Heike Hegewald, Hashem Yazbek

Software Engineering Group
Institute for Distributed Systems
University of Magdeburg, Germany

{dumke, makunz, yazbek}@ivs.cs.uni-magdeburg.de
http://ivs.cs.uni-magdeburg.de/sw-eng/us/

Abstract. This paper concludes a description of an agent-based measurement support system in order to install a measurement process based on ISO/IEC standard 15939. The different tasks and subjects of the software agents have been developed to represent the constituents of that standard itself. A first prototype is described and some future applications and extensions are discussed.
The Agent RelaTEd Mesurement InfraStructure (ARTEMIS) is an example how an agent-oriented software engineering approach leads us from monolithic applications to flexible, reliable and autonomously acting agent-based systems as a new kind of measurement infrastructures.

Analysis of the Designs of Coupling Measures: A Case Study

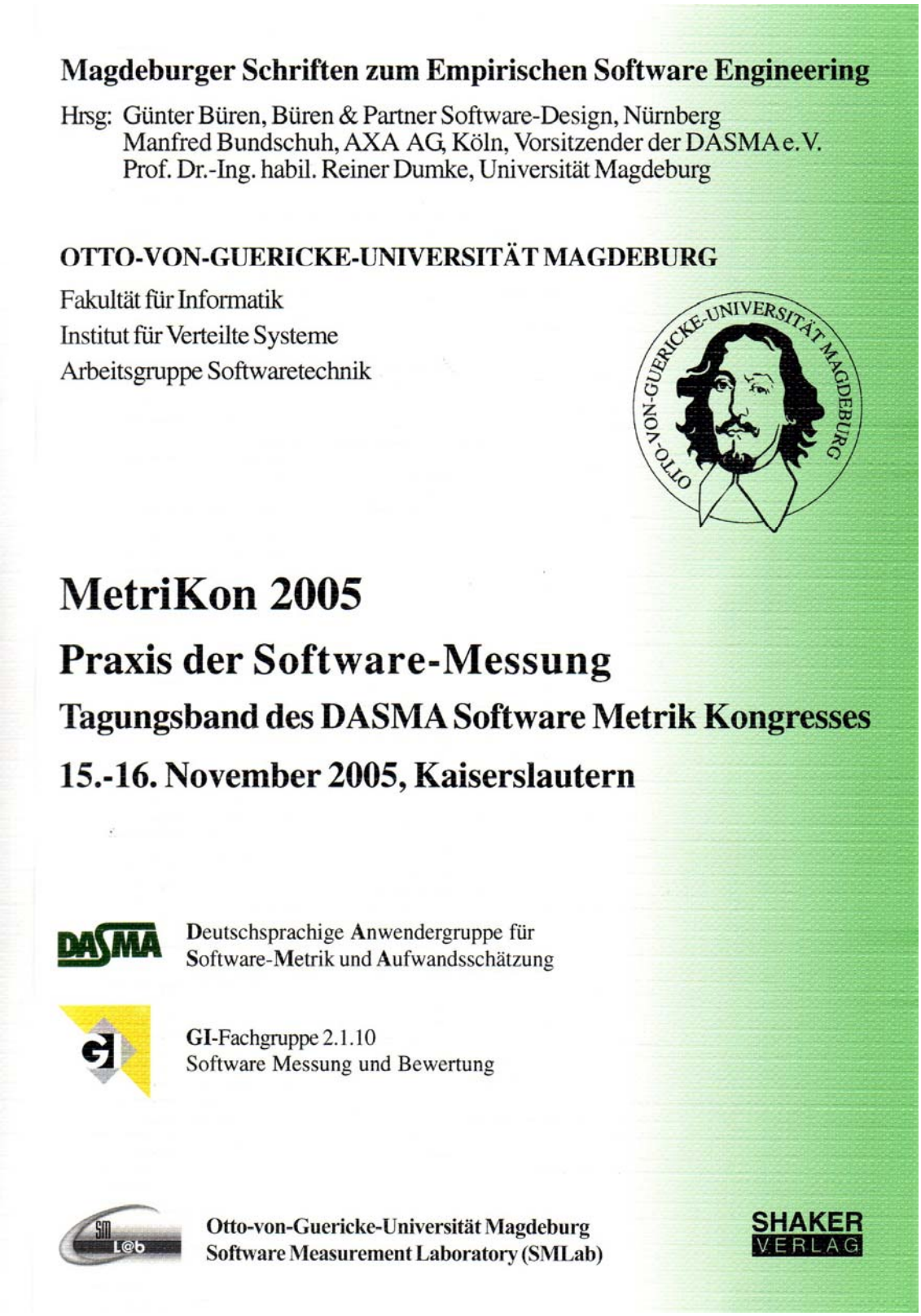
Laila Cheikhi[1], Alain Abran[1], Miguel Lopez[2]

[1]Ecole de Technologie Supérieure, Montréal, Québec; Canada
laila.cheikhi.1@ens.etsmtl.ca, alain.abran@.etsmtl.ca

[2]CETIC asbl, Rue Clément Ader, 8, B-6041 Gosselies, Belgium
mlo@Cetic.bc

Abstract. Various measures have been proposed in software engineering for evaluating the quality of object-oriented software systems, many of them aimed at measuring the structural properties of the design of the software, such as coupling, cohesion and inheritance. There is diversity among the current proposals for coupling measurements and models, reflecting a lack of consensus on coupling and the need for a reference framework. This paper investigates the design of coupling measures based on Abran and Jacquet's model of the process for designing a measurement method. This analysis is illustrated with a case study using measures of one type of coupling, suggested by Chidamber and Kemerer: Coupling Between Objects (CBO). This case study verifies whether or not this CBO measure includes all the design elements of a measurement method.

Our DASMA Software Metrik Kongress (MetriKon 2005) took place in Kaiserslautern, Germany in November 2005. The following report gives an overview about the presented papers. Furthermore, the papers are published in the following Shaker book (ISBN 3-8322-4615-0):



Magdeburger Schriften zum Empirischen Software Engineering

Hrsg: Günter Büren, Büren & Partner Software-Design, Nürnberg
Manfred Bundschuh, AXA AG, Köln, Vorsitzender der DASMA e.V.
Prof. Dr.-Ing. habil. Reiner Dumke, Universität Magdeburg

OTTO-VON-GUERICKE-UNIVERSITÄT MAGDEBURG

Fakultät für Informatik
Institut für Verteilte Systeme
Arbeitsgruppe Softwaretechnik

MetriKon 2005

Praxis der Software-Messung

Tagungsband des DASMA Software Metrik Kongresses

15.-16. November 2005, Kaiserslautern

DASMA Deutschsprachige Anwendergruppe für Software-Metrik und Aufwandsschätzung

GI-Fachgruppe 2.1.10
Software Messung und Bewertung

Otto-von-Guericke-Universität Magdeburg
Software Measurement Laboratory (SMLab)

SHAKER VERLAG

KEYNOTE:

Software-Qualitätsmessung - Von der Theorie zur Empirie

Prof. Dr. Peter Liggesmeyer

Fraunhofer-Institut für Experimentelles Software Engineering

Zusammenfassung. Viele Unternehmen verfügen heute über definierte Prozesse, die jedoch oft noch nicht unter quantitativer Kontrolle sind. Die Einführung und erfolgreiche Nutzung von Messsystemen ist für diese Unternehmen daher der nächste Schritt.
Die Messung von Software-Qualität - z.B., des Fehlergehalts - ist direkt nicht möglich. Die Ansätze zur Bestimmung der Software-Qualität bedienen sich daher statistischer Verfahren zur Auswertung von Beobachtungen, z.B. aus dem Systemtest, oder basieren auf Schlussfolgerungen auf Basis direkt messbarer Eigenschaften.
Ein einschlägiges Messsystem ist nur dann sinnvoll einsetzbar, wenn es gelingt, die erforderlichen Schlüsse mit ausreichender Präzision zu ziehen. In diesem Zusammenhang kommt empirischen Studien, die theoretische Überlegungen praktisch validieren, eine hohe Bedeutung zu.
Im Rahmen des Vortrags werden die Ansätze zur Software-Qualitätsmessung diskutiert und Ergebnisse ausgewählter empirischer Untersuchungen aus diesem vorgestellt.

Usability Metriken – Stand und Perspektiven

Christina Zahn, Ralf Ueberfuhr, Rüdiger Liskowsky

Technische Universität Dresden,
Fakultät Informatik, Lehrgebiet Programmierumgebungen und Werkzeuge

master@webtini.de, master@ru-soft.de, rl2@inf.tu-dresden.de

Zusammenfassung. Es wird versucht, die vorhandenen Ansätze zur Messung der Gebrauchstauglichkeit (Usability) von Software systematisch zu beschreiben. Ausgegangen wird von bekannten Standards und Normen, um auf einer soliden Basis aufzubauen. Die weiteren Ausführungen stützen sich auf eine gründliche Analyse bekannter Usability-Werkzeuge und ihrer Messfunktionen, woraus eine umfangreiche hierarchische Gliederung des gegenwärtigen Standes genutzter Usability Metriken abgeleitet wird. Die aufgefundenen Metriken und die benötigten Werkzeugfunktionen bilden den Ausgangspunkt für die Entwicklung eines eigenen universellen Usability-Messwerkzeuges, das die Technische Universität Dresden zunächst zu Versuchszwecken zur Verfügung stellen will. Die weiteren Perspektiven von Metriken für die Gebrauchstauglichkeit werden hauptsächlich vom Gesichtspunkt der zukünftigen Werkzeugentwicklung gesehen.

Verwendung der Data Envelopment Analysis im Kontext von ERP Implementierungsprojekten: Vergleich und Aufwandsschätzung

Stefan Koch

Institut für Informationswirtschaft, Wirtschaftsuniversität Wien

stefan.koch@wu-wien.ac.at

Zusammenfassung. In den ketzten Jahren hat der Bereich der Enterprise Resource Planning (ERP) Systeme wie beispielsweise SAP R/3 eine enorme Bedeutung für die Software-Branche erlangt. Insbesondere die damit einhergehenden Einführungsprojekte zeichnen sich durch hohe Komplexität und damit verbunden ein hohes Risiko aus. In diesem Artikel wird die Data Envelopment Analysis (DEA) als Werkzeug vorgeschlagen, um sowohl den Vergleich solcher Projekte hinsichtlich Effizienz, aber auch eine Aufwandsschätzung neuer Projekte zu ermöglichen. Da ERP-Einführungsprojekte sich von „traditioneller" Software-Entwicklung, die vor allem in der Generierung von Programmcode mündet, unterscheiden, erweist sich die DEA, die mit mehrdimensionalen, unterschiedlich skalierten Inputs und Outputs umgehen kann, als tauglicher Ansatz. Dies wird über die Demonstration beider Verwendungszwecke anhand eines Datensatzes von 39 Einführungsprojekten aus einer Umfrage unter österreichischen Unternehmen unterstrichen.

Eine Untersuchung zum Metrikeinsatz in der Industrie

Miriam Fink, Tilmann Hampp

Institut für Softwaretechnologie, Universität Stuttgart
Universitätsstr. 38, 70569 Stuttgart

finkmm@swt.uni-stuttgart.de, hampp@informatik.uni-stuttgart.de

Zusammenfassung. In der Software-Entwicklung wurde in den letzten Jahren das Thema Metriken immer wichtiger, nicht zuletzt durch die zunehmende Bedeutung von CMM und CMMI. Ziel dieser Arbeit war es, den praktischen Einsatz von Metriken in der Software-Industrie empirisch zu untersuchen. Hierbei stehen die folgenden Fragen im Vordergrund:„Welche Metriken werden tatsächlich eingesetzt?", „Welche Probleme werden mit diesen Metriken angegangen?" und „Welcher Nutzen wird aus diesen Metriken gezogen?". Die Untersuchung ergab, dass Metriken primär zur Projektkontrolle eingesetzt werden. Für die Qualitätssicherung scheinen Metriken nur zur Testkontrolle weiter verbreitet zu sein. Metriken zur Programmstruktur werden kaum eingesetzt.

Empirische Betrachtungen zur Softwareentwicklung im Rahmen von Offshore-Kooperationen

Andreas Schmietendorf#, Reiner Dumke+*

T-Systems GmbH, Entwicklungszentrum Berlin – eSC/IS, Wittestraße 30H, 13509 Berlin
andreas.schmietendorf@t-systems.com

* Fachhochschule für Wirtschaft Berlin, FB II, Neue Bahnhofstraße 11-17, 10245 Berlin
a.schmietendorf@ba-berlin.de

+ Otto-von-Guericke-Universität Magdeburg, Fakultät Informatik - IVS,
Postfach 41 20, 39016 Magdeburg
dumke@ivs.cs.uni-magdeburg.de

Zusammenfassung. Die industrielle Softwareentwicklung bedient sich zunehmend der Leistungen so genannter Offshore-Anbieter in Niedriglohnländern. Primäres Ziel der Offshore-Entwicklung ist die Kostensenkung während der Implementierung und damit die Erhaltung der Konkurrenzfähigkeit auf dem Markt gegenüber anderen Anbietern. Dennoch steht hinter dem Begriff des Offshoring weit mehr als nur eine bloße Kostensenkung. Für eine langfristige erfolgreiche Zusammenarbeit gilt es, vielfältige Einflußkriterien zu berücksichtigen. Der Begriff des Offshoring wird zumeist differenziert in Offshore und Nearshore. Nearshore bezieht sich aus zentraleuropäischer Sicht auf die Auslagerung von Dienstleistungen ins (ost-)europäische Ausland. Offshore bezieht sich auf die Auslagerung von Aufgabenstellungen in den asiatischen Raum. Im Rahmen dieses Beitrags wollen wir nach der Vorstellung theoretischer Grundlagen zum Offshoring insbesondere auf praktische Erfahrungen eingehen, die während einer nunmehr 6-jährigen Kooperation mit einem bulgarischen Partner, also dem Nearshoring, gewonnen werden konnten.

M-System NT - Ein flexibles, datenbank-basiertes Mess- und Analyse-System

Jürgen Münch, Axel Wickenkamp

Fraunhofer-Institut für Experimentelles Software Engineering

juergen.muench@iese.fraunhofer.de
axel.wickenkamp@iese.fraunhofer.de

Zusammenfassung. Wesentliche Anforderungen an Messansätze bei der Entwicklung von Software und software-intensiven Systemen sind Zielorientierung, Zweckgebundenheit und Kontextabhängigkeit. Die Bedeutung dieser Anforderungen wird insbesondere im Zusammenhang mit kontinuierlichern Verbesserungsansätzen (wie z.B. dem Quality Improvement Paradigm, Profes, Six Sigma) deutlich, deren Modellunabhängigkeit ein besonderes Maßschneidern von Messansätzen erfordert. Auf methodischer Ebene werden diese Anforderungen durch den Goal-Question-Metric-Ansatz (GQM) abgedeckt. Auf der Ebene von Messwerkzeugen beschränkt sich die Unterstützung bisher oft auf die Erfassung vordefinierter Maße mit geringen Anpassungsmöglichkeiten. Gerade auf Kodeebene ist es vielfach notwendig, unter Ausnutzung syntaktischer und semantischer Spracheigenschaften maßgeschneiderte Metriken zu definieren, werkzeuggestützt zu erfassen und zu analysieren. Im Rahmen des Beitrags werden das methodisch gestützte Werkzeug M-System NT zur Analyse und Messung statischer Eigenschaften von Quelltexten, dessen initiale Erprobung sowie Erfahrungen mit dem System vorgestellt.

Goal-oriented Data Visualization with

Software Project Control Centers

Jens Heidrich, Jürgen Münch

Fraunhofer Institute for Experimental Software Engineering

jens.heidrich@iese.fraunhofer.de
juergen.muench@iese.fraunhofer.de

Abstract. Many software development organizations still lack support for obtaining intellectual control over their software development processes and for determining the performance of their processes and the quality of the produced products. Systematic support for detecting and reacting to critical project states in order to achieve planned goals is usually missing. One means to institutionalize measurement on the basis of explicit models is the development and establishment of a so-called Software Project Control Center (SPCC) for systematic quality assurance and management support. An SPCC is comparable to a control room, which is a well known term in the mechanical production domain. Its tasks include collecting, interpreting, and visualizing measurement data in order to provide context-, purpose-, and role-oriented information for all stakeholders (e.g., project managers,
quality assurance manager, developers) during the execution of a software develJens Heidrich, Jürgen Münch Software Metrik Kongress opment project. The article will present an overview of SPCC concepts, a concrete instantiation that supports goal-oriented data visualization (G-SPCC approach), and experiences from practical applications.

QScope - Metriken formulieren, berechnen und visualisieren

Daniel Germanus, Lukas Mrokon

Fachbereich Informatik
Technische Universität Darmstadt

daniel.germanus@gmail.com, lukas.mrokon@web.de

Zusammenfassang. QScope ist eine Sammlung mehrerer aufeinander abgestimmter Plugins für die frei verfügbare Entwicklungsumgebung Eclipse zur Berechnung von Softwaremetriken. Gezielte Eingrenzung und schnelles Erkennen wichtiger Bereiche in Softwareprojekten setzen eine Visualisierung und eine weit gefächerte Erweiterbarkeit voraus, um unter stetig veränderten Randbedingungen diskretisieren zu können.
Auf die Übersicht der verwendeten Technologien folgt ein Abschnitt über die Abfragesprache XQuery mit deren Hilfe Metriken formuliert werden können. Die verschiedenen Komponenten von QScope werden im Detail vorgestellt und eine Aussicht auf zukünftige Leistungsmerkmale präsentiert. QScope ist frei verfügbar, es unterliegt einer Open-Source-Lizenz und entstand im Rahmen eines Software Engineering Praktikums an der TU-Darmstadt.

An ISO 15939-Based Infrastructure Supporting the IT Software Measurement

Reiner Dumke, Renè Braungarten, Martin Kunz, Heike Hegewald

Otto-von-Guericke-University of Magdeburg, Germany

(dumke,braungar,makunz)@ivs.cs.uni-magdeburg.de

Abstract. This paper concludes a description of an agent-based measurement support system in order to install a measurement process based on ISO/IEC standard 15939. The different tasks and subjects of the software agents have been developed to represent the constituents of that standard itself. A first prototype is described and some future applications and extensions are discussed.
The Agent RelaTEd Mesurement InfraStructure (ARTEMIS) is an example how an agent-oriented software engineering approach leads us from monolithic applications to flexible, reliable and autonomously acting agent-based systems as a new kind of measurement infrastructures.

Resolving the Mysteries of the Halstead Measures

Horst Zuse

TU-Berlin, Franklinstraße 28/29, 10587 Berlin
Phone: +49-30-881 59 88, Fax: +49-30-886 81 678

horst.zuse@t-online.de
http://www.zuse.info

Abstract. The Halstead measures are still a subject of mystery. The properties of these measures are not well understood and there is still a confusion of the scales (types). Halstead's measures are used by many tools, but the usefulness and the meaning of the numbers of these measures are not clear. We use the concept of the extensive structure from measurement theory in order to investigate and resolve the mysteries of the Halstead measures.

cGQM - Ein zielorientierter Ansatz für kontinuierliche, automatisierte Messzyklen

Christoph Lofi

Collaborative Software Development Laboratory, University of Hawai'i Manoa

christoph.lofi@gmail.com

Zusammenfassung. Messungen, als Grundlage für einen vorhersagbaren und kontrollierbaren Software-Entwicklungsprozess, sind ein wichtiger Teilaspekt des Software Engineering. Das Durchführen von Messungen ist essentiell zum Bewerten des aktuellen Projektstatus, dem Erfassen von „Baselines" (Ausgangsmesswerte) und der der Validation von Verbesserungs- und Kontrollaktionen.
Die in diesem Artikel vorgestellten Arbeiten basieren auf Hackystat, einem vollautomatisiertem Messwerkzeug für Software Produkt- und Prozessmetriken. Hackystat wurde um die Fähigkeit der zielgerichteten Messung und Analyse nach dem GQM (Goal, Question, Metric) Paradigma erweitert.
Die so entstehende Messplattform implementiert ein ebenfalls hier vorgestelltes Konzept namens cGQM (continuous GQM), einer Spezialisierung des GQM Paradigmas mit der Einschränkung auf lediglich automatisch erfass- und analysierbare Metriken. cGQM zusammen mit seiner Referenzimplementierung hackyCGQM stellen einen interessanten, vollautomatisierten Ansatz zur werkzeuggestützten Softwareprojektkontrolle dar.
Die aus diesem Ansatz entstehenden Vor- und Nachteile werden in diesem Artikel vorgestellt und kurz diskutiert.

Design eines wertorientierten Metriksystems für die Projektsteuerung im Rahmen von Software-Projekten

Evgenia Wollenhaupt

Hochschule Pforzheim, Siemens AG

e.wollenhaupt@web.de

Zusammenfassung. Diese Arbeit beschäftigt sich mit der Konzipierung eines wertorientierten Metriksystems für die Projektsteuerung im Rahmen von Software-Projekten bei der Siemens AG. Systematisch erfassten und beschriebenen Zeit-, Aufwands- und Sachfortschrittsmetriken sowie integrierte Analysen bilden eine aussagekräftige Basis dafür. Die Zusammenhänge zwischen der Projektmessung und dem Einsatz bestimmter Vorgehensmodelle werden ebenfalls untersucht.
Die Entwicklung eines Metriksystems soll auf spezifischen Verbesserungszielen und Prozessbesonderheiten basieren, damit das entwickelte Metriksystem Erfolgsaussichten hat und einen Nutzen mit sich bringt.

Measuring the Effectiveness of Software Testing (Converting Software Testing from an Art to a Science)

Harry M. Sneed

Software Test Engineer
ANECON GmbH, Vienna, Austria
Universities of Regensburg & Passau, Bavaria

Harry.Sneed@Anecon.com

Abstract. The proposed paper presents a set of metrics developed by the author while working as a test consultant for a Viennese software house from 1998 until 2003. They are intended to measure and predict the performance of a test operation. With these metrics it should be possible to convert software testing from an art as perceived by Glenford Meyers in 1975 to a science as defined by Lord Kelvin in 1875. The metrics were obtained using the Goal Question Metric Method of Basili and Rombach and were refined through practical application and empirical study. They are supported by a set of tools designed for both static and dynamic analysis as well as for evaluating the results of both.

Project Management in New Domains through Process-oriented Collection and Analysis of Effort Data

Fabio Bella, Alexis Ocampo, Jürgen Münch

Fraunhofer-Institut für Experimentelles Software Engineering (IESE)

{bella, ocampo, muench}@iese.fraunhofer.de

Abstract. The planning and control of software development projects in new domains such as the emerging Field of wireless services represent a special challenge: On the one hand, new technologies open up possibilities whose constraints are unclear• at the beginning of such a project; an the other hand, there is no proven knowledge regarding domain-specific problems and risks as well as the effort needed to deal with those. Some of the consequences to be expected are unreliable project planning, incorrect effort estimation, and unsuitable risk management with respect to the processes, resources, and technologies to he used.

This article describes how the descriptive process modeling approach can be combined with process-oriented collection and analysis of effort data for identifying domain-specific problems and risks. This combined approach is illustrated with examples from real projects. Benefits as well as potential difficulties encountered with the approach are discussed.

Definition and evaluation of system requirements metrics based on CMMI

Silvia Linschi[1], Marek Leszak[2]

[1]Friedrich-Alexander University Erlangen-Nürnberg, Faculty of Business Administration, Lange Gasse 20, 90403 Nürnberg, Germany
silvialinschi@web.de

[2]Lucent Technologies Network Systems GmbH, Optical Networking Group, Thurn-und-Taxis-Str. 10, 90411 Nürnberg, Germany,
mleszak@lucent.com

Abstract. This paper presents the results of an industrial case study, aiming at defining and evaluating CMMI conformant metrics for requirements development and requirements management, for the Lucent product LambdaUnite™ MSS. This product is a large-scale, embedded (hardware/software) digital transmission multiplexer system for the metropolitan and wide-area telecommunications market, being developed evolutionary since 2001 in multiple customer releases. Within our case study, state-of-the-art requirements metrics have been investigated, selected, and extended by own, newly defined metrics. These metrics have been specified in detail, measured, validated and analyzed. Most defined metrics have been successfully deployed within our Systems Engineering organization, for better monitoring and improving the system requirements work. To the authors' best knowledge, very few studies on industrial measuring the requirements management process have been published, so there are rather few related studies to compare our results.

Die Requirement Points Analyse - ein Ansatz zur Aufwandsschätzung im Requirements Engineering

Nicolas Fernando Porta

Universität Ulm, Abteilung Angewandte Informationsverarbeitung (SAI)

nicolas_porta@web.de

Zusammenfassung. Dem Requirements Engineering als jener Disziplin der Informatik, die die Erstellung und Pflege von Spezifikationen umfasst, kommt angesichts der düsteren Statistik von ITProjekten eine wichtige Rolle zu. Ihm obliegt es, Verfahren zur Verfügung zu stellen, die qualitativ hochwertige und vollständige Anforderungsdokumente ermöglichen. Um über geeignetes Reporting im Requirements Engineering verfügen zu können, werden Schätzverfahren benötigt, die unmittelbar auf die Phase der Spezifikationserstellung zielen. Während die meisten gängigen Verfahren entweder IT-Projekte in ihrer Gesamtheit schätzen, oder Anforderungen als Input verwenden, um Aussagen über die Systemgröße zu treffen, möchte dieser Beitrag einen anderen Weg gehen. Durch die Bewertung bereits vorhandener Spezifikationsdokumente in Requirement Points kann der zu erwartende Aufwand ähnlicher Spezifikationen durch Analogie-Schätzung bestimmt werden. Die Requirement Points Analyse definiert hierzu Klassen von Anforderungen sowie syntaktische und semantische Kriterien, nach denen Spezifikationen eingeteilt werden. Zusätzlich werden Einflussfaktoren bestimmt und von einzelnen Anforderungen auf die Gesamtspezifikation aggregiert. Die Requirement Points Analyse liefert damit eine Grundlage für Aufwandsschätzungen im Requirements Engineering und ermöglicht darüber hinaus eine Reihe von Kennzahlen, die den Status der Spezifikation wiedergeben.

Software-Metriken als Schlüssel zur Aufwandsabschätzung - Anforderungsmanagement in der Automotive SW-Entwicklung

Dr. Dirk Abendroth, Wolfram Bohne

BMWAG München, Antriebsentwicklung

Dirk.Abendroth@bmw.de,Wolfram.Bohne@bmw.de

Zusammenfassung. Mit dem Ziel, die Aufwände für das Anforderungsmanagement für alle Funktionen einer Motorsteuerung zu bestimmen, wird in diesem Beitrag eine Metrik für Software-Funktionen entwickelt. Die vorgestellte Metrik ist basierend auf der ASCET1-Notation formuliert, die im Automotive Bereich als High Level-Beschreibung der Funktionssoftware verwendet wird.
Das Ergebnis der Kapazitätsabschätzung entsteht durch das Zusammenwirken zweier Informationsteile: Zum einen müssen die relevanten Charakteristika des betrachteten Software-Umfanges in eine adäquate Metrik übersetzt werden. Im Spannungsfeld des Automotive SW-Engineering gilt es in dieser Metrik ebenso den Umfang einer Software als auch deren Vernetzung und Komplexität in geeigneter Weise zu berücksichtigen.
Zum anderen muss die abgeleitete Software-Metrik in Aufwände übersetzt werden, die im Anforderungsmanagement für diese Funktionen ausgelöst werden. Zu diesem Zweck sind auf breiter fachlicher Basis und auf verschiedenen Abstraktionsebenen Pilotprojekte zum Anforderungsmanagement durchgeführt worden. Für die aus der in den Pilotierungen entstandende Funktionssoftware werden die Metrikwerte und die Anforderungsmanagement-Aufwände gegenübergestellt.
Aus der Schätzung der Metrik für den Gesamtumfang der Software-Funktionen und dem Übersetzungsverhältnis aus den Pilotierungen wird dann der Gesamtaufwand des Anforderungsmanagement für alle Funktionen einer Motorsteuerung extrapoliert.

Nicht falsch, sondern das Falsche Geschätzt! Warum viele Function Point Installationen scheitern.

Robert Hürten

Hürten & Partner Unternehmensberatung, Nippes 2, 53945 Blankenheim

robert.huerten@huerten-partner.de

Zusammenfassung. In der Mehrzahl der Unternehmen, die sich vor Jahren mit viel Aufwand mit der Einführung der Function Point Methode beschäftigten, nutzen diese heute nicht mehr. Als häufigste Gründe für diese Entwicklung werden genannt:

- Wir erhielten keine zufriedenstellenden Schätzungen.
- Programmiertätigkeit ist kreativ und somit nicht meßbar.
- Die Function Point Analysis stellt zu hohe Ansprüche an die Dokumentation.
- Der Aufwand steht in keinem Verhältnis zum Erfolg.

Konsequenz:

- Wir erhielten keine zufriedenstellenden Schätzungen.

Die Function Point Analysis wird fälschlicherweise als eine Methode zum Schätzen des Aufwandes von Softwareentwicklungsaktivitäten vermittelt und verkauft.
Den wenigsten Anwendern ist bewusst, dass Function Points nur ein Maß für einen der vielen Kostentreiber in der Softwareentwicklung ist. Der Aufwand resultiert aus einer Vielzahl von Kostentreibern.

Messen des Projekt-Risikos

Dipl.-Ing. Johannes Bergsmann

Allgemein gerichtlich beeideter und zertifizierter Sachverständiger für Informatik
Staatlich befugter und beeideter Ingenieurkonsulent für Informatik
Gründer und Eigentümer von Software Quality Lab / Österreich

johannes.bergsmann@software-quality-lab.at

Zusammenfassung. Das Management von Projektrisiken kann als integraler Bestandteil eines guten Projektmanagements angesehen werden.
Mit Hilfe des Risikomanagements werden mögliche Gefahren identifiziert, bewertet und entsprechende Maßnahmen und Reaktionen definiert.
Dieser Beitrag soll das oft unterschätzte Thema Projektrisiken sensibilisieren und Methoden und Möglichkeiten aufzuzeigen, um Projektrisiken zu definieren, zu berechnen und in monetäre Faktoren umzurechnen.
Durch eine adäquate Risikoanalyse soll eine möglichst objektive Entscheidungsgrundlage über eine Fortführung oder den Abbruch des Projekts geschaffen werden.
Im Falle einer Entscheidung zur Fortführung des Projekts kann durch eine systematische Vorgehensweise auch eine methodische Unterstützung der Projektleitung bei der Überwachung und Steuerung der Projektrisiken geboten werden.
Dieser Beitrag stellt unter anderem auch die Methode PARisQ (Preliminary Advanced Riskanalysis Quantitative) für die Risikoanalyse in Software-Projekten vor, die speziell den schwierigen Bereich der Risikoidentifikation und monetären Bewertung von Projektrisiken betrachtet und zeigt dabei auch weitere Problempunkte bei der Risikoanalyse auf, auf die besonders geachtet werden sollte.

Ermittlung von Synergiepotenzialen auf Grundlage der Function Point Analyse

Dipl.-Math. Jürgen Bach, Dr. Marcus Mauser

Bundeskanzlerplatz 2-10, 53113 Bonn (bach consulting GmbH)

Juergen.Bach@bachconsulting.de, Marcus.Mauser@bachconsulting.de

Zusammenfassung. In den vergangenen Jahren ist ein steigendes Interesse an einer realistischen Einschätzung von Synergieeffekten in verschiedensten Bereichen der IT zu beobachten. Dabei greifen rein kostenorientierte Synergiebetrachtungen oftmals zu kurz, funktionsorientierte Aspekte werden dagegen zu wenig berücksichtigt.
Ausgangspunkt ist die kritische Reflexion der Begriffe Synergie, Kosten und Funktionalität. Die oft überschätzte Erwartung an sog. Synergiepotenziale wird durch eine definierte und messbare Größe ersetzt, die als Kriterium akzeptabel und als Entscheidungsmoment praktikabel ist. Die Autoren folgen dabei ihrer bewährten Trichotomie: *Definieren - Messen - Entscheiden*.
Der Artikel entwirft ein umfassendes Modell, bei dem aus der zeitlichen Zuordnung von Funktions- und Kostenbestandteilen ein Synergiepotenzial für einen Produkt-Life-Cycle abgeleitet wird. Funktionalitäten sind durch Expertensysteme projektklassenspezifisch in Kosten überführbar und ein Vergleich der sich ergebenden Synergiedarstellungen erscheint interessant.
Zur Bestimmung des funktionsorientierten Synergiepotenzials bei der Integration zweier IT-Produkte ist es notwendig, die identifizierten Funktionen nach Funktionsgruppen zu klassifizieren. Für die Ermittlung der Funktionalität von ITProjekten sind in der Vergangenheit verschiedene Metriken entwickelt worden, wobei sich die Function Point Analyse (FPA) als besonders effektiv erwiesen hat.

Vom betriebswirtschaftlichen Interesse sind die Life-Cycle-Kosten eines Produkts. Bereits in der Vorprojektphase kann durch FPA mit hinreichend hoher Genauigkeit auf die Entwicklungs-, sowie die Produktions- und die Rollout-Kosten geschlossen werden. Mit Hilfe eines projektspezifischen, empirischen Verteilungsschlüssels werden die zu erwartenden Life-Cycle-Kosten eines IT-Produkts abgeleitet. Damit ist ein kosten- und nutzenorientiertes Synergie–potenzial entlang der Projektphase darstellbar.
Anhand konkreter Beispiele wird das Synergiemodell erläutert, wobei zusätzlich der Sättigungseffekt bei langen Zeitdauern, die Berechnung des Break-Even-Points sowie das Konzept der kritischen Kosten, kritischen Zeitpunkte und kritischen Funktionsumfänge diskutiert werden.
Der Schwerpunkt liegt hier auf der Betrachtung von IT-Produkten; der Beitrag will aber auch Anregung für weitergehende Applikationen sein. So scheint die zugrunde liegende Logik des messbaren Synergiepotenzials (Modell und Formel) auch geeignet, andere Aggregationsstufen zu bedienen. Prinzipiell sollte es möglich sein, von der Funktionalität bis zu dem Unternehmenswert Anwendungsmöglichkeiten zu erschließen.
Die Zukunft wird zeigen, ob aus der FPA abgeleitete, messbare Synergiepotenziale auch bei der Bewertung von IT-Produktportfolios oder gar bei Entscheidungen über Fusionen bzw. Kooperationen von IT-Unternehmen dienlich sein können.

Warum Prüfen oft 50 mal länger dauert als Lesen und andere Überraschungen aus der Welt der Software-Reviews

Dipl.-Inform. Peter Rösler

freiberuflicher Trainer für Reviewtechnik, München

ros@reviewtechnik.de

Zusammenfassung. In Schulungen zu Software-Reviews steht der Trainer immer dann vor einer didaktischen Herausforderung, wenn es darum geht, die von Fachleuten genannten Zahlenangaben zu begründen, die von den Teilnehmern intuitiv in ganz anderer Größenordnung eingeschätzt werden.
Zwei dieser Zahlenangaben, die den Teilnehmern üblicherweise besonders unplausibel vorkommen, sind:

1. Die optimale Inspektionsrate für Textdokumente beträgt nur ca. 1 Seite pro Stunde (und liegt damit um ca. den Faktor 50 unter der reinen Lesegeschwindigkeit).
2. Das durchschnittliche Review findet nur ca. 5% der im Dokument vorhandenen Fehler.

Im Beitrag wird gezeigt, mit welchen Kurzexperimenten und Schätzungen, die von den Teilnehmern selbst durchgeführt werden, obige Zahlenangaben zumindest in der Größenordnung als durchaus plausibel dargestellt werden können.

Das wundersame Verhalten von Entwicklern beim Einsatz von Quellcode-Metriken

Daniel Simon, Dr. Frank Simon

SQS Software Quality Systems AG, Stollwerckstraße 11, 51149 Köln

daniel.simon@sqs.de, frank.simon@sqs.de

Zusammenfassung. Dieser Erfahrungsbericht gibt einen Einblick in häufig beobachtete Verhaltensweisen von Entwicklern in Software-Projekten, bei denen aus unterschiedlichen Gründen ein kennzahlenbasiertes Qualitätsmanagement für Quellcode etabliert werden soll. Die von den Entwicklern empfundene Skepsis bei der Einführung neuer, für sie häufig ungewohnter Technologien in Form von Quellcode-Metriken muß allerdings für deren effektiven Einsatz zuvor überwunden werden. Jede der aufgeführten fünf Abwehr-Haltungen bedarf hierfür eigener Überzeugungs-Strategien. Diese helfen, ein durch die Entwickler selbst getragenes Vorgehen zu etablieren, in dem allen Beteiligten klar ist, daß auch die Entwickler selbst von einer transparenten Qualität des eigenen Quellcodes profitieren, und sind damit Grundvoraussetzung dafür, dass der Einsatz von Quellcode-Metriken maximalen Mehrwert bringen kann.

Benchmarking is an essential control mechanism for management

Ton Dekkers

Sogeti Nederland B.V.

ton.dekkers@sogeti.nl

Abstract. A great number of big organisations already has been outsourcing the IT activities or is thinking about outsourcing. Even outsourcing companies in some cases use specific variances in outsourcing IT, e.g. near-shore or offshore. But are these activities so beneficial, are they introducing other risks or is it all opportunity?
Management needs to find a way to decide on transparent and objective criteria whether outsourcing could be beneficial. First of all a measurement model for controlling outsourcing or investigating the cost benefit ratio for outsourcing should be adopted. Key issue in most of these models is project delivery rate. To be able to compare and to judge project delivery rates, benchmarking is a good way to resolve this topic.

KEYNOTE:

Produktivität messen und verbessern

Dr. Christof Ebert

Alcatel, Paris

Zusammenfassung. Produktivität ist eine betriebswirtschaftliche Schlüsselzahl, denn sie beeinflusst direkt, was an einem Projekt verdient werden kann. Trotzdem fristet sie im Softwaregeschäft nach wie vor eine Nischendasein. Das liegt weniger daran, dass man mit Software auch ohne Produktivität Geld verdienen kann, als an der Angst, das falsche zu messen.

Über Jahrzehnte haben uns die Gurus eingeimpft, dass Codezeilen über Aufwand als Produktivitätsmaß nicht hinreichend sind - aber sie haben auch niemals ein besseres Maß geliefert. So indoktriniert steuern wir praktisch alle Projekte nach Gutdünken und akzeptieren, dass der Aufwand zwar von der Produktivität abhängt, man diese aber nicht messen kann.

Diesen gordischen Knoten will die Keynote durchschlagen. Sie setzt ganz klar darauf dass man Produktivität messen und verbessern muss. Zu warten, bis uns andere Unternehmen die Butter vom Brot nehmen, ist eine Vogel-Strauß-Strategie, die uns sehr bald aus dem Geschäft wirft.

Speziell betrachtet die Keynote Produktivitätsmessung mit drei Zielen: Projektschätzungen verbessern, Lieferanten bewerten, Produktivität verbessern. Es werden einzelne konkret nutzbare Produktivitätsindikatoren vorgestellt, die praktisch eingesetzt werden können - und mit entsprechendem Einsatz auch zu Verbesserungen der Produktivität führen.

Harmonization Issues in the Updating of ISO Standards on Software Product Quality

Alain Abran, Rafa E. Al-Qutaish, Jean-Marc Desharnais

École de Technologie Supérieure (ÉTS)
University of Québec
1100 Notre-Dame Street West
Montréal, Québec H3C 1K3, Canada

alain.abran@etsmtl.ca, rafa.al-qutaish.1@ens.etsmtl.ca, jean-marc.desharnais@etsmtl.ca

Abstract. *Within the context of the current ISO project to upgrade the set of technical reports on the measurement of the quality of software products (ISO 9126), the ISO working group concerned has come up with proposals for various documents (standards or technical reports) in the new ISO 25000 series to improve the interpretation and use of the quality measures. This paper investigates some of the harmonization issues arising with the addition of new documents like ISO 25021, in particular with respect to previously published measurement standards for software engineering, such as ISO 9126, ISO 15939, ISO 14143-1 and ISO 19761.*

Keywords: Software Product Quality, Software Measurement, ISO 25021, ISO 9126, ISO 15939.

1 Introduction

In 1991, the ISO published its first international consensus on the terminology for the quality characteristics for software product evaluation (ISO 9126:1991) [1]. During the period 2001 to 2004, the ISO published an expanded version, containing the ISO quality models and a consensus on inventories of proposed measures for these models. The current version of the ISO 9126 series of standards consists of four documents [2-5]:

- ISO 9126-1: Quality Models
- ISO TR 9126-2: External Metrics[1]
- ISO TR 9126-3: Internal Metrics
- ISO TR 9126-4: Quality in Use Metrics

The ISO has now recognized a need for further enhancements to ISO 9126, primarily as a result of advances in the field of information technologies and changes in environment [6]. Therefore, the ISO is now working on the next generation of software product quality standards, which will be referred to as Software Product

[1] The term 'metrics' used in ISO 9126 is replaced by 'measures' in the new series of standards, in accordance with ISO 15939.

Quality Requirements and Evaluation (ISO 25000). This series of standards will replace the current ISO 9126 and ISO 14598 series, and will consist of five divisions [7], each of which may contain one or more documents:

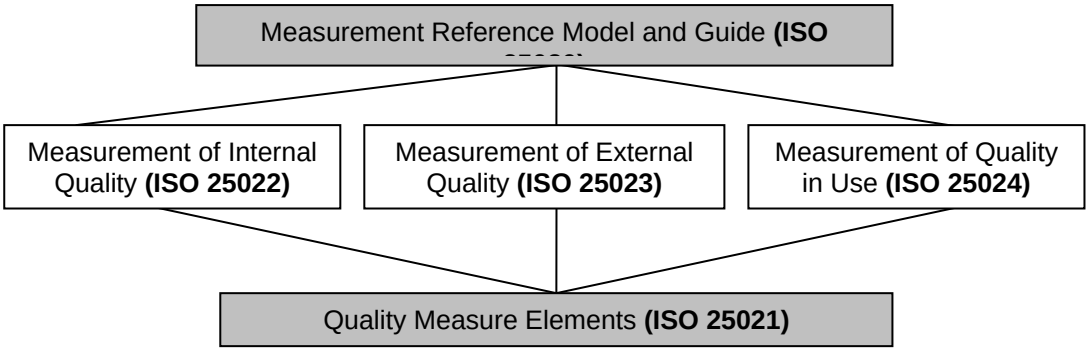
- ISO 2500n: Quality Management Division
- ISO 2501n: Quality Model Division
- ISO 2502n: Quality measurement Division
- ISO 2503n: Quality Requirements Division
- ISO 2504n: Quality Evaluation Division

This work is being carried out by Working Group 6 (WG6) of the software and system engineering subcommittee (SC7) of the ISO/IEC joint technical committee (JTC1) on Information Technology, that is, ISO/IEC JTC1/SC7.

One of the objectives of this new ISO 25000 series (and what makes it different from the current ISO 9126 series) is the harmonization of its contents with the software measurement terminology of ISO 15939 [8], itself based on the ISO metrology terminology [9]. Figure 1 shows the proposed structure of the quality measurement division (ISO 2502n) series that is to replace the current four-part ISO 9126 series of standards [10]. This proposed quality measurement division (ISO 2502n) will consist of five documents:

- ISO 25020: Measurement Reference Model and Guide
- ISO 25021: Quality Measure Elements
- ISO 25022: Measurement of Internal Quality
- ISO 25023: Measurement of External Quality
- ISO 25024: Measurement of Quality in Use

Figure 1: WG6 Proposed Structure of the Measurement Division (ISO 2502n series)



Included in this new set of technical reports is a proposed new structure with additional new concepts, such as: ‘quality measure elements’ (QME) and ‘software quality measures’ [10]. This paper also investigates these proposed concepts, their

use and interpretation, and their relationship to similar concepts in other ISO documents.

This paper discusses the issues concerning terminology harmonization in section 2, and the issues concerning the harmonization of quality model coverage between ISO DTR 25021 and ISO 9126 in section 3. A discussion, conclusions and recommendations are presented in section 4.

2 Terminology harmonization

2.1 Metrology terminology

The ISO 9126 working group (WG6) has proposed the introduction of four new expressions in ISO DTR 25021 [10], namely: ‘Quality Measure Elements’, ‘General Quality Measure Elements’, ‘Specific Quality Measure Elements’ and ‘Quality Measures’. The introduction of these new terms raises the following concern: either the proper mapping to the set of classic metrology terms has not yet been completed or there are concepts and related terms missing in the metrology vocabulary. The latter would be surprising, since metrology is a rather mature domain of knowledge based on centuries of expertise in the field of measurement and related international standardization. In this paper, we revisit the WG6 proposal in order to recommend a proper mapping of concepts to the related metrology [9] terms and to ISO 15939 [8].
The following two expressions come from the ISO standard on software measurement process, ISO/IEC 15939 [8], which is itself based on the definitions in the ISO International Vocabulary of Basic and General Terms in Metrology (VIM 1993) [9]:

Base measure: a measure defined in terms of an attribute and the method for quantifying it. A base measure is functionally independent of other measures.

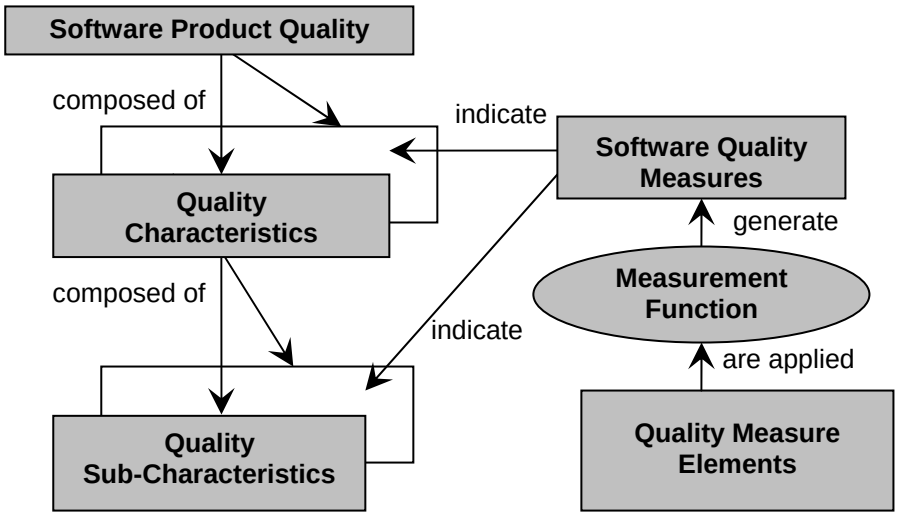
Derived measure: a measure defined as a function of two or more values of base measures. A transformation of a base measure using a mathematical function can also be considered as a derived measure.

In [10], it is claimed that a quality measure element is either a base measure or a derived measure, but then the consensual metrology terms are ignored in favor of locally defined WG6 measures, thus bypassing the ISO and SC7 harmonization requirements on measurement terminology.

The ‘quality measure elements’ are described as an input for the measurement of the ‘software quality measures’ of external quality, internal quality and quality in use [10]. Figure 2 shows the proposed relationship between the ‘Quality Measure Elements’ and the ‘Software Quality Measures’, and between the ‘Software Quality Measures’ and the quality characteristics and subcharacteristics. In metrology, these would correspond to base measures and derived measures respectively. It can be observed as well that these measures, in particular the derived measures, are defined specifically to measure the subcharacteristics of internal and external quality or the

characteristics of quality in use. None of these is directly related to the top level of ‘software quality’ (which is itself decomposed into three models, then into 16 characteristics and further into a large number of subcharacteristics). Therefore, the expression selected, in [10], ‘software quality measures’,, is at a level of abstraction that does not represent the proper mapping of the measures to the concept being measured.

Figure 2: Quality Measure Elements Concept in the "Software Product Quality Measurement Reference Model" [10]



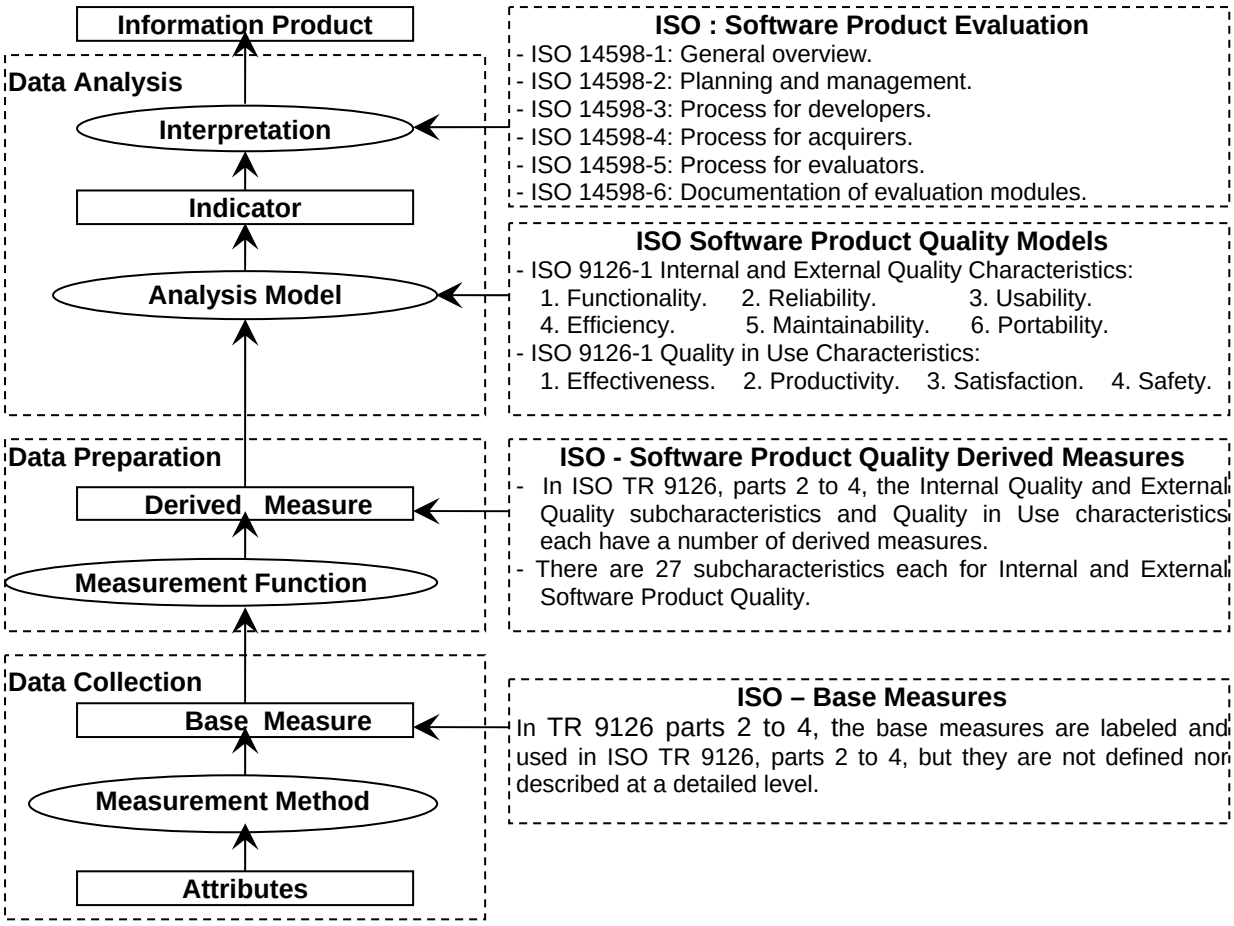
2.2 Harmonization with the ISO 15939 Information Model

The ISO 15939 information model has been divided into three different sections: data collection, data preparation and data analysis [11]. Figure 3 shows a mapping between this information model (left-had side of Fig. 3) and the software product quality measurement and evaluation of ISO series 9126 and 14598 (right-hand side of Fig. 3).

2.3 Description Harmonization

The WG6 proposal in [10] recommends next a set of 15 ‘General Quality Measure Elements’ – Table 1 – to be used as ‘Specific Quality Measure Elements’ within the software product life cycle; [10] includes a description of its selection of 61 such ‘Specific Quality Measure Elements’. However, it is noted that there are no specific quality measure elements related to the general quality measure elements ‘Number of User Operations’ or ‘Number of System Operations’.

Figure 3: Mapping between ISO 15939 Information Model and ISO 9126 and ISO 14598



For the description of each of these quality measure elements, different ‘aspects’ are proposed in [10]:

1. Scale type: the aspect related to the scale type used for measurement.
2. Focus: the aspect related to the scope and objective of the measurement (e.g. the software product itself, the software product in a system, the software product in a system used by a specified user in a specified scenario).
3. Method type: the aspect related to the measurement method type relating to the quality measure element used for measurement.

For aspect 3, it is stated in the same document that the scope and objective are related to the different parts of ISO 9126 (internal quality, external quality and quality in use). It must be noted that the use of expressions such as ‘scope’ in [10] for a measurement method is not harmonized with the corresponding ‘scope’ terminology used in other ISO software measurement-related standards, such as ISO 14143-1 [12] and ISO 19761 [13].

Table 1: ‘General Quality Measure Elements’ [10]

1. Number of Functions	2. Number of Failures
3. Number of Faults	4. Product Size
5. Time Duration	6. Number of Test Cases
7. Number of Restarts	8. Number of I/O
9. Number of Trials	10. Number of Data Items
11. Data Size	12. Number of Requirements
13. Number of Tasks	14. Number of User Operations
15. Number of System Operations	

It can also be observed that, in Table 1, a number of the quantities have a label starting with ‘number of’. However, these do not use a reference scale typical of measures in the sciences or in engineering, but are rather counts on entities. For any of these proposed counts, such as the ‘number of functions’, no specific method is proposed for an identification of the number of functions in a consistent manner across measurers and organizations; for instance, the definition of the word ‘function’ could differ from one individual to another within the same organization, and more so across organizations. Therefore, to say in [10] that such numbers are obtained by an ‘objective’ method is an overstatement, since they must be obtained mostly on the basis of the judgment of the person carrying out the count.

Of the 15 proposed general quality measure elements, only ‘time’ comes from a classic base measure using, for instance, the international standard unit of the second (or a multiple or submultiple of it) as its reference scale. There are also measuring instruments to ensure that time measurements are indeed obtained in an objective manner.

It can also be observed that, of the 15 measures proposed in Table 1, at most four are directly related to the quality of software: number of faults, number of failures, number of restarts and number of trials. None of the other 12 measures is directly or indirectly related to the quality of software. In fact, they are strictly independent of it per se,, as they are often used for normalization purposes, for instance.

Finally, in [14], the issue of documenting a base measure using the full set of metrology concepts about quantities and units has been investigated and recommendations provided that would lead to a more comprehensive design of the software measure.

2.4 Lack of reference to corresponding ISO measurement standards

For the ‘product size’ general quality measure element, [10] lists many ways to measure product size: lines of code, function points, modules, classes and visual structures. There are also various methods for counting lines of code and for measuring function points. Therefore, this general quality measure element could be further split into different base measures. Moreover, the ISO has specified mandatory requirements for function point measurement methods [12], and has recognized four different functional size measurement methods as ISO standards meeting these requirements, such as COSMIC-FFP [13]. None of these existing ISO software

engineering standards, which are referenced in ISO 90003 [15], has been mentioned or referenced in [10]. Also, the various methods available to obtain those numbers have their strengths and weaknesses, from a measurement perspective, in terms of repeatability, reproducibility, software domains of applicability and accuracy.

3 Coverage Harmonization

3.1 Limited coverage of the ISO quality models and corresponding measures

ISO TR 9126, parts 2 to 4, presents the ISO inventory of measures for the full coverage of the ISO software product quality models (internal quality, external quality and quality in use) for measuring any of their quality characteristics and subcharacteristics. The full sets of base measures in these three parts of ISO 9126 are presented in Appendix A and include 82 base measures.

Of these 82 base measures, only 15 are included in [10]; this means that the coverage in [10] is very limited, and the reasons for this are not obvious. The proposed content coverage of this subset of base measures is claimed in [10] to be the 'most important'; however, no specific criteria to determine its 'importance' are provided. Some generic information is provided in [10] to suggest that these measures were derived from a questionnaire-based survey; however, it does not provide the reader with information about the criteria for selection, the size and representativeness of the sample in the countries where the data were collected, or the representativeness of this sample outside these countries. Another claim, that "*they represent a default kernel of quality measures, which are proven to be beneficial and common practice*" [10], is not supported by documented evidence, nor is there a discussion of its generalizability outside its data collection context.

Appendix B presents a detailed analysis of the coverage of the quality measures in [10], together with the corresponding availability in ISO TR 9126, parts 2 to 4. Appendix B specifically illustrates that 15 measures for the 'internal quality' of software product are selected in [10] out of an inventory of 70 in the corresponding ISO TR 9126-3, while 55 measures are excluded, again without a documented rationale.

Furthermore, the 15 measures of internal quality selected in [10] cover only 4 of the 6 quality characteristics of the ISO model of internal quality, and only 9 of 27 subcharacteristics; again, the rationale for excluding any characteristic or subcharacteristic is not documented.

Similarly, for the 'Quality in Use' quality measures, [10]:

- Includes only 2 quality measures of the 15 already available in ISO TR 9126-4
- Excludes 2 QIU characteristics, that is, 'safety' and 'satisfaction'
- Does not include any Specific Quality Measure Elements related to the 'Number of User Operations' and 'Number of System Operations'

3.2 Overlapping issues

Some additional information included in [10] has already been covered in ISO TR 9126 documents, and will be included in the ISO 25000 series; for instance, information about the 'scale types' is covered through rephrasing information contained in other documents, once again increasing synchronization and harmonization right away and over the long term. Similarly for the narratives about the measures of internal software quality, external software quality and software quality in use, as well as for the narratives about the software measurement methods.

This is contrary to the ISO practice of avoiding duplication or the rephrasing of information across ISO documents, and increases the possibility of inconsistencies across documents; it could later lead to significant effort over the long term in maintaining synchronization of documents covering similar subsets of information.

These examples point to configuration management issues over the long term which will represent additional cost to the purchasers of these ISO documents, since they will be required to pay twice for the same information which is a subset of the full inventory. This could lead to some confusion for standards users as to which of these documents is most valuable to a standard purchaser, and under what circumstances.

4 Discussion

4.1 Summary of harmonization issues in ISO DTR 25021

The ISO is now working on the next generation of software product quality standard, which will be referred to as Software Product Quality Requirements and Evaluation (ISO 25000). One of the objectives of this new ISO 25000 series (and what differentiates it from the current ISO 9126 series) is the harmonization of its contents with the software measurement terminology of ISO 15939 [8], itself based on the ISO metrology terminology [9]. In this paper, terminology harmonization issues have been identified, as well as the coverage of harmonization issues in ISO DTR 25021 and ISO 9126 in terms of the coverage of ISO quality models.

Below is a summary of the harmonization issues identified:

A) Terminology in [10]:

- what is referred to as a 'quality measure element' corresponds to the classic concept of 'base measure' in ISO 15939;
- what is referred to as 'software quality measure':
    - corresponds to the classic concept of 'derived measure' in ISO 15939;
    - is not at the proper level of abstraction for the concept being measured when mapped to the hierarchy of concepts for software product quality adopted by the ISO.

B) Harmonization with the Information Model of ISO 15939:

- unless the terminology is harmonized with ISO International Vocabulary of Basic and General Terms in Metroogy, then it is challenging to align the older versions of the ISO 9126 and ISO 14598, and it will be even more challenging with the upcoming updates in ISO 25000.
- should the harmonization of terminology be accepted, it becomes then easier to map each of these ISO 9126 and 14598 series into the Information Model of ISO 15939.

C) Description harmonization:

- A large number of the base measures proposed are counts of entities rather than measures per se with required metrological characteristics, such as: unit, scale, dimension, measurement method, measurement procedures, etc.
- In [10], in some instances, like ‘product size’ for example, there is no reference to other existing ISO standards for software size, such as ISO 19761, etc.
- There are a number of claims that the proposed base measures are ‘objective’, while they are obviously derived from a manual process without precisely documented measurement procedures, thereby leaving much to the measurer’s judgment.

D) Coverage harmonization in [10]:

- The set of base measures documented represents only a limited subset of the base measures within ISO 9126, parts 2 to 4; the rationale for inclusion or exclusion is not documented.
- The set of base measures does not allow coverage of the full spectrum of quality characteristics and subcharacteristics in ISO 9126, parts 2 to 4; again, the rationale for inclusion or exclusion is not documented.

4.2 Recommendations

From the above analysis, the following recommendations are put forward:

- Ensure that the terminology on software product quality measurement is fully aligned with the classic measurement terminology in the sciences and in engineering;
- Provide full coverage of the base measures for all three ISO models of software quality;
- Provide improved documentation of the base measure using the criteria from metrology;
- Provide clear mapping and traceability of the new ISO 25000 documents to the ISO 15939 Information Model.

Acknowledgments

The opinions expressed in this paper are solely those of the authors.

References

[1] ISO/IEC, ISO/IEC IS 9126, Software Product Evaluation - Quality Characteristics and Guidelines for Their Use, Geneva, Switzerland: International Organization for Standardization, 1991.

[2] ISO/IEC, ISO/IEC 9126-1: Software Engineering - Product Quality - Part 1: Quality Model, Geneva, Switzerland: International Organization for Standardization, 2001.

[3] ISO/IEC, ISO/IEC TR 9126-2: Software Engineering - Product Quality - Part 2: External Metrics, Geneva, Switzerland: International Organization for Standardization, 2003.

[4] ISO/IEC, ISO/IEC TR 9126-3: Software Engineering - Product Quality - Part 3: Internal Metrics, Geneva, Switzerland: International Organization for Standardization, 2003.

[5] ISO/IEC, ISO/IEC TR 9126-4: Software Engineering - Product Quality - Part 4: Quality in Use Metrics, Geneva, Switzerland: International Organization for Standardization, 2004.

[6] M. Azuma, "SQuaRE: The next Generation of ISO/IEC 9126 and 14598 International Standards Series on Software Product Quality," Proceedings of the European Software Control and Metrics Conference (ESCOM), London, UK, 2001. pp. 337-346.

[7] ISO/IEC, ISO/IEC 25000: Software Engineering - Software Product Quality Requirements and Evaluation (SQuaRE) - Guide to SQuaRE, Geneva, Switzerland: International Organization for Standardization, 2005.

[8] ISO/IEC, ISO/IEC IS 15939: Software Engineering - Software Measurement Process, Geneva, Switzerland: International Organization for Standardization, 2002.

[9] ISO/IEC, International Vocabulary of Basic and General Terms in Metrology (VIM), Geneva, Switzerland: International Organization for Standardization, 1993.

[10] SC7, ISO/IEC DTR 25021: Software Engineering - Software Product Quality Requirements and Evaluation (SQuaRE) - Quality Measure Elements ISO/IEC JTC1/SC7 WG6, January 11, 2006, 6N-565, 2006.

[11] A. Abran, R. E. Al-Qutaish, J. M. Desharnais, and N. Habra, "An Information Model for Software Quality Measurement with ISO Standards," in Proceedings of the International Conference on Software Development (SWDC-REK), Reykjavik, Iceland, 2005. pp. 104-116.

[12] ISO/IEC, ISO/IEC 14143-1: Software Measurement - Functional Size Measurement - Part 1: Definition of Concepts, Geneva, Switzerland: International Organization for Standardization, 1998.

[13] ISO/IEC, ISO/IEC 19761: Software Engineering - COSMIC-FFP - A Functional Size Measurement Method, Geneva, Switzerland: International Organization for Standardization, 2003.

[14] A. Abran, R. Al Qutaish, and J. Cuadrado, "Investigation of the Metrology Concepts within ISO 9126 on Software Product Quality Evaluation," submitted to the fourth Workshop on Software Quality - 28th International Conference on Software Engineering - ICSE 2006, Shanghai, China, July 2006.

[15] ISO/IEC, ISO/IEC 90003: Software Engineering - Guidelines for the Application of ISO 9001:2000 to Computer Software, Geneva, Switzerland: International Organization for Standardization, 2004.

Quality assurance of service development projects for service oriented architectures

Andreas Schmietendorf

University of Cooperative Education at
Berlin School of Economics
Neue Bahnhofstrasse 11-17, D-10245 Berlin, Germany

andreas.schmietendorf@ba-berlin.de

1 Introduction

As through [8] described every major company that wants to survive within a global marketplace has to change in an e-business company. The change to an e-business company implies at least the following aspects:

- Changed business processes (high degree of automation).
- Definition of an IT-architecture that supports e-business capabilities.
- Adjusted procedures for the software (service) development.
- Changed procedures for the operating of IT-applications.
- Intensive utilization of the possibilities of the Internet.

Especially the Web services technology supports this goal. Web services are attractive for industrial software development, and play an important role in the field of integration solutions and the current challenge of establishing service-oriented architectures (short SOA). A Web service can described as piece of software located in the Internet, whose public interface is described using WSDL (Web Service Description Language) and that uses XML-based protocols (e.g. SOAP, Simple Object Access Protocol) to communicate with clients and other services. Web service metadata can be published in a registry (e.g. UDDI, Universal Description, Discovery and Integration), where it can be found by probable clients in order to establish a connection with the service.

To establish service oriented architectures it requires rules and guidelines for the “step by step” implementation of service offerings and a procedure for a business process driven integration of this service offerings. Furthermore it needs quality assurance systems for service development projects. Currently, there are only few approaches for the quality assurance of software development projects in the field of service oriented architectures. But those projects differ relatively strong from classical software development projects. This means those new business requirements are implemented through the integration of already existing systems and their offered services, as on the basis of new developed applications.

This paper describes an ongoing work. After this short introduction it describes the basics and principles of a service-oriented architecture. Furthermore it investigates

the possible contents of a guideline for the service development, respectively the service integration. In addition, some aspects are introduced for the use of measurements. Finally the paper gives a conclusion and proposes necessary research projects.

2 SOA at a glance

Like mentioned through [Dostal 2004], the idea of SOA is not only driven from the Web service technology. Basic concepts were already available in the context of distributed communication mechanism like DCE or CORBA. Over it out SOA considers process relevant aspects, required resource and the product itself.

2.1 Basics of Service oriented architectures

Before talking about service-oriented architectures (SOA), the concepts of software architecture shall be clarified by a rather classic definition of [2]:

> *”The software architecture of a program or computing system is the structure or structures of the system, which comprise software components, the externally visible properties of those components, and the relationships among them.”*

Nevertheless, currently we have no common definition for understanding the concepts of SOA. From my view, the development happens less on technological level. Also aspects of the IT- and business-architecture, but also semantic questions are important to reach a SOA. The following definition of SOA, pointed out by the Gartner Group supports this thesis:

> *“Essentially, SOA is a software architecture that builds a topology of interfaces, interface implementations and interface calls. SOA is a relationship of services and service consumers, both software modules large enough to represent a complete business function. Services are software modules that are accessed by name via an interface, typically in a request-reply mode. Service consumers are software that embeds a service interface proxy (the client representation of the interface).”*

Generally, service-oriented architectures can be characterized by the fact that they separate the implementation of the service from its interface. Withal, a ”find, bind, and execute” paradigm enables a service’s customer to query a third-party registry for an adequate service implementation. In case the registry finds a matching service, it provides the customer with a contract and an endpoint address. The hardware, the used operating system or the used programming language to the implementation of the service doesn't play any role on that occasion. Following the notes of [10], SOA configures its six entities, namely service consumers, providers, registries, contracts, proxies, and service leases after all, to support the above mentioned paradigm.

A service is a function that is well-defined, self-contained, and does not depend on the context or state of other services.

- well-defined
    - a service provides well described functionalities
    - the access of functionalities is only possible though the interface
- self contained
    - a service is completed in itself
    - a service hides implementation details (black box view)
- not depend on the context or state of other services
    - independently from the context and conditions of other services
    - decoupled with reference to the place, the protocol and time of other services

2.2 Principles of Service offerings

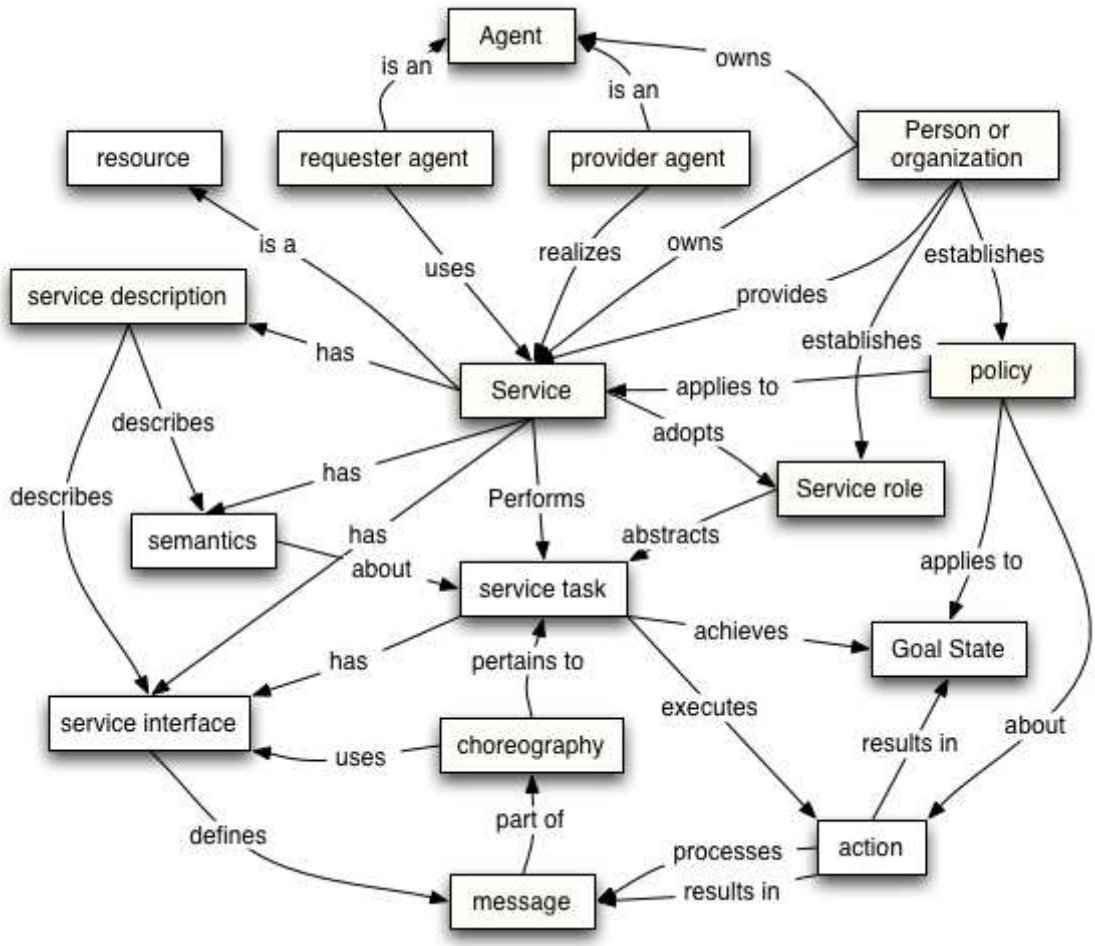
To describe the principles of service offerings, I want to use the architecture approach from World Wide Web consortium (short W3C). This approach provided by [3] considers different aspects of Web service based architectures. These aspects deal with the:

- Service Oriented Model
- Message Oriented Model
- Policy Model
- Resource Oriented Model

All together these models support the implementation of a service-oriented architecture under the consideration of the Web service technology. This paper concentrates on the "Service Oriented Model". This model shown in figure 1 identifies the relationships to all important aspects of a service offering. Typical examples are the service description or the performed service tasks.

After the architectural view from the W3C, the view provided by [7] will be shown. He proposes the following principles in context of service-oriented architectures. Services are reusable, Services share a formal contract, Service are loosely coupled, Services abstract underlying logic, Services are composable, Services are autonomous, Services are stateless and Services are discoverable. Furthermore he pointed out that autonomy, loose coupling, abstraction and the need for a formal contract form the baseline foundation for a SOA.

Figure 1: Architecture-Model of a service from view of the W3C



3 Goals and structure of a service guideline

Service-oriented architectures and service offerings implies a high complexity for the necessary development- and integrations-tasks. In big enterprises such a goal position can be reached only through the establishing of internal standards (service guideline). The question is which aspects should take into account of a corresponding service guideline. To answer this question it is necessary to consider the mentioned principles of service offerings from the previous chapter and specifics of projects for service oriented architectures. For this a service guideline should consider at least the following aspects:

- Development of service offerings
- Procedures for the integration of services to support new business processes
- Quality assurance inclusive requirements for test activities
- Recommendation for the deployment and management

3.1 Related works

At this point I want to show known approaches from the literature and the industries. The first one proposes the use of a SOA Maturity Model [13]. Like the well known Capability Maturity Model (CMM) and its successor CMM Integration (CMMI) [4] from the Software Engineering Institute (SEI), the SOA Maturity Model proposes the use of five maturity levels. It describes for each level the business benefits, the scope of service-oriented solutions, technological success criteria and organizational success criteria. Furthermore, Web service related standards are assigned to the individual levels. In the individual one the levels can be characterised as follows:

1. Stating with basics services for new functionalities by the use of standards like XML, XSLT, WSDL, SOAP, Java, .NET
2. Architecture related services (for cost reduction and cost control) by the use of standards like UDDI, WS-Policy, WS Adressing, WS-Security
3. A: Enterprise related services (flexible reaction to changes within business processes), by the use of standards like WS-BPEL
3. B: Collaboration related services (Collaboration with business partners), by the use of standards like RosettaNet, ebXML, WS-Trust
4. Measurable enterprise services (real time enterprise)
5. Optimized services (automated reaction of new requirements)

An interesting article about the establishing of a service-oriented architecture can be found under [9]. He tries to answer the “six most common questions” from Chief Information Officers, Chief Technology Officers and Chief Architects. In the individual, the paper considers the following questions:

1. How do we get started with SOA? Where do we begin?
2. What Services should we begin with? How do we identify the appropriate Services for our initial SOA projects?
3. What SOA technology solutions do we need and in what sequence?
4. What governance model and policies do we need? Who “owns” the SOA efforts?
5. How do we measure SOA results? What’s the ROI?
6. What organizational, process and skills issues will we face?

Furthermore [9] pointed out the following important statement.

> *“SOA is not a quick fix for all of your business or IT challenges, but it does provide a clear strategic pathway forward for your organization. Remember, the IT challenges that SOAs address are artefacts of years of corporate decisions and behaviours over time. It will take time to undue the accumulated technological complexity in your IT architecture.”*

An approach for the implementation of a service design guideline can be found by [7]. This approach considers primary developer related aspects. He proposes to consider

naming standards (endpoint names, operation names, message values), suggestions for a suitable level of interface granularity, the extensibility of service operations, the use of modular wsdl-documents and potential service requestors.

This observation emphasizes the demand of a strategic procedure to the “step by step” introduction of a SOA. Such procedure must strategic, administrative, and operative tasks take into account.

3.2 Content of a service guideline

At first a service guideline shows the migration strategy from the current architecture to a SOA. Such a guideline should consider technical aspects, business implications and resource related requirements too. Furthermore a service guideline should support the development of a service offering and the procedure of service integration to fulfil new business requirements.

The following contents should be contained:

1. Overview to entire IT-architecture of the enterprise and their development during the next years. This overview shows internal and external “Back End” applications and provided interfaces. Furthermore this overview must consider the integration architecture, like the used Enterprise Service Bus, required functionalities of the service interfaces, used process and/or workflow engines and service management capabilities. For a good understanding of the current situation it is necessary to show the relationships between the IT-systems and corresponding business processes.

2. Tasks during the development of service offerings oriented at the classical phases of software development.
   - Analysis of the concerned business processes, business functionalities and corresponding business objects for each business functionalities. During this phase it is necessary to map a process-related model to a service-related model. This task designation decides about the granularity and employable ness of the service implementation. Specification of the required messages per service and definition of simple and complex data types. These results provide the basis for the XML-specification.
   - The design considers the data model of the used applications (wrapping of functionalities). The goal is to specify a corresponding XML-specification and to map the data model from the application to the service specification. Further tasks consider potential constraints, the transactional behaviour and potential error cases.
   - During the phase of implementation the development of the Web service is carry out. Besides the required functionalities, the Web Service should provide further operation for the version and configuration management, the transaction-management and the logging-management. Further activities consider a test framework and the deployment within a runtime-equivalent environment.

3. The integration of services is necessary to fulfil new business requirements. These assembling must be supported by an integrations framework. The framework supports modelling and simulation capabilities and provides a link to used process and/or workflow-engines. A service guideline should show possible application scenarios for specific service-offerings and supported business processes respective business functionalities.

4. Another important aspect of the guideline should be the quality assurance aspects. Quality aspects deal with the used test framework, test requirements, used measurements or the service specification. The quality behaviour of a service during the runtime can be characterized by the use of service policies (see chapter 2.2).

5. Furthermore the guideline must provide a section were the developer can find a reference development environment and important aspects of the runtime environment (e.g. used application servers and the supported version). Especially the last aspect is important, because the service implementation must run within a standardized environment.

4 Metrics-based evaluation of service offerings

Well known is the fact that the use of measurements supports the quality assurance of software development projects. The question is what kind of measurements should be used within projects for service oriented architectures. [6]

4.1 Introductory comments

[14] pointed out that typical application development metrics (such as function points, or number of lines of code or classes) are no longer useful in measuring the progress of developers in an SOA-project. He proposed to establish a new set of metrics for measuring the performance of developers and projects in an SOA. Under consideration of a service oriented paradigm, used metrics should consider such aspects like:

- Level of reused services
- Kind of integration for the realization of a new process
- Reached degree of the automation
- Granularity and modularity of service offerings
- Description of the service interface
- Reaction time until the implementation of a new request

An interesting approach for the description of software and services can be found in [Böttcher 2004]. He pointed out, that the complexity of a service depends not from the covered problems or structures, as more from the number of possible conditions which a service can have during a given time period.

Regarding the application of measurements, it must be distinguished between the service development and the service integration. Within the implementation phase of a new Web service we can use measurements, from the classical field of software development, like metrics for the object-oriented software development (e.g. number of classes, inheritance deep, complexity measures for a method). Important is the consideration of the specific behaviour of the service interface. For example this interface must provide descriptions for the functional and the non-functional characteristics. A first approach for an evaluation of Web service interfaces can be found in [12]. It considers such aspects like the functional description form the developer's viewpoint, pre- and post conditions, process-related aspects and semantic relationships.

4.2 Industrial SOA-projects

For the identification of required measurements it is useful to analyse real SOA-projects from the industries. As I mentioned within the introduction this paper deals with an ongoing work. Therefore I want to describe the aims of the quality assurance tasks within an industrial SOA-project. Currently it is impossible to show final results. The investigated project deals with the wrapping of existing information systems. The functionalities of the legacy applications are accessed by the use of the Java Remote Methods Invocation Interface (RMI) technology. This requires the use of a Java-container as run-time environment. A so called controller application works as mediation device between the RMI-clients and the Web service interface implementation.

It provides the following functionalities:

- Offer of a Web service based interface
- Controlling of the business process execution
- Message based event- and notification-handling
- Functionalities for logging and recovery

The Web service supports 13 business oriented functions. For the support of these functions the Web service divided direct operations and indirect operations.

- Size of the wsdl-file for the direct operations: 2117 LoC
- Size of the wsdl-file for the indirect operations: 1820 LoC

All together we can count 41 operations, which require the use of 82 SOAP-messages. Within each SOAP-message it is necessary to provide meta-information to support a specific error handling. This meta-information is stored in a database and can used in the case of errors, timeouts or transaction management.

5 Summary and further works

In this paper I introduced some results of an ongoing investigation. The aims of this work deal with the identification of necessary quality assurance activities within SOA-

projects. Currently there are many research activities within this topic, like the following ones:

- Process definition of the whole quality assurance activities within a SOA-project
- Measurement tools for process-, resource- and product related SOA-aspects
- Well tested pattern for analysis-, design-, implementation-, test- and deployment-tasks
- Empirical analysis of service development projects
- Empirical analysis of service integration projects
- Model driven approaches for SOA-projects

Furthermore the implementation of an SOA-suitability evaluation model is planned. Such an evaluation model covers aspects like the well-defined functionalities of a service offering; the self-contained behaviour - a service is completed in itself and hides implementation details; the service does not depend on the context or state of other services; the granularity features - the service provides the right number of business functionalities. The service composition or orchestration is another important aspect within the planned evaluation model.

References

[1] Alonso, G.; Casati, F.; Kuno, H.; Machiraju, V.: Web Services – Concepts, Architectures and Applications, Springer-Verlag, Berlin, 2004

[2] Bass, L.; Clements, P.; Kazman. R.: Software Architecture in Practice. Addison-Wesley Professional, 2nd (hardcover) edition, 2003.

[3] Booth, D.; Haas, H.; McCabe, F.; Newcomer, E.; Champion, M.; Ferries, C.; Orchard, D.: Web Service Architecture. NOTE-ws-arch-20040211, W3C – World Wide Web Consortium, Boston/MA, 2004

[4] Software Engineering Institute, “Capability Maturity Model Integration” (CMMI), Version 1.1, Carnegie Mellon University, Pittsburgh, PA, USA, Tech. Rep. CMU/SEI-2002-TR-012, March 2002

[5] Dostal, W.; Jeckle, M.; Melzer, I.; Zengler, B.: Service-orientierte Architekturen mit Web Services (Konzepte – Standards – Praxis), Spektrum Akademischer Verlag, München, 2005

[6] Ebert, C.; Dumke, R.; Bundschuh, M.; Schmietendorf, A.: Best Practices in Software Measurement - How to use metrics to improve project and process performance. Springer, Berlin Heidelberg, 2005.

[7] Erl, T.: Service-Oriented Architecture – Concepts, Technology and Design, Prentice hall, Upper Saddle River, NJ, 2005

[8] Harmon, P.; Rosen, M.; Guttmann, M.: Developing E-Business Systems and Architectures – A Managers’s Guide, Morgan Kaufmann Publ., San Francisco/USA, 2001

[9] Marks, E. A.: Answering the Six Most-Asked SOA Questions - An AgilePath Point of View, http://www.agile-path.com/, 2005

[10] McGovern, J.; Tyagi, S.; Stevens, M. E.: Java Web Services Architecture, Morgan Kaufmann, 2003.

[11] Newcomer, E.; Lomow, G.: Understanding SOA with Web Services, Person Education, Upper Saddle River, NJ, 2004

[12] Schmietendorf, A.; Dumke, R.: Empirical Analysis of available Web Services, in Proc. of IWSM 2003

[13] SOA MATURITY MODEL, Der Leitfaden für die Einführung einer serviceorientierten Architektur, Sonic Software Corporation, AmberPoint Inc., BearingPoint, Inc., Systinet Corporation.

[14] Vecchio, D.; Smith, D.: Service-Oriented Development - A Future View, Gartner European Symposium, Cannes/France, 2005

Thanks

I would particularly like to thank my customer Mr. Koch for his support in this work as it was he who made the work possible in the first place. Furthermore my thanks go to Prof. Dr. Reiner Dumke for the stimulating discussions within this topic.

Büren, G.; Bundschuh, M.; Dumke, R.:

MetriKon 2005 – Praxis der Software-Messung

Shaker Verlag,Aachen, November 2005 (299 Seiten)
ISBN 3-8322-4615-0

The book includes the proceedings of the DASMA Metric Conference **MetriKon 2005** held in Kaiserslautern in November, 2005, which constitute a collection of theoretical studies in the field of software measurement and case reports on the application of software metrics in companies and universities.

The contents are described by the listing of the paper abstracts in this Metrics News.

Abran, A.; Dumke, R.:

Innovations in Software Measurement

Shaker Verlag, Aachen, September 2005 (456 Seiten)
ISBN 3-8322-4405-0

The book includes the proceedings of the 15th International Workshop on Software Measurement (IWSM2005) held in Montreal in September, 2005, which constitute a collection of theoretical studies in the field of software measurement and case reports on the application of software metrics in companies and universities in Argentina, Australia, Austria, Bahrain, Belgium, Brazil, Bulgaria, Canada, Finland, France, Germany, Ghana, Italy, Netherlands, Poland, Slovenia, Spain, Switzerland, UK, USA and Vietnam.

The contents are described by the listing of the paper abstracts in this Metrics News.

Kandt, R.K.:

Software Engineering Quality Practices

Auerbach Publications, 2006 (256 Seiten)
ISBN 3-8493-4633-9

Software Engineering Quality Practices describes how software engineers and the managers who supervise them can develop quality software in an effective, efficient, and professional manner. This volume conveys practical advice quickly and clearly while avoiding the dogma that surrounds the software profession. It concentrates on what the real requirements of a system are, what constitutes an appropriate solution, and how you can ensure that the realized solution fulfils the desired qualities of relevant stakeholders. The book also discusses how successful organizations attract and keep people who are capable of building high-quality systems.

The author succinctly describes the nature and fundamental principles of design and incorporates them into an architectural framework, enabling you to apply the framework to the development of quality software for most applications. The text also analyzes engineering requirements, identifies poor requirements, and demonstrates how bad requirements can be transformed via several important quality practes.

Ebert, C.:

Systematisches Requirements Management Anforderungen ermitteln, spezifizieren, analysieren und verfolgen

dpunkt.verlag, August 2005 (320 Seiten)
ISBN 3-89864-336-0

Projekte scheitern häufig wegen unzureichendem Requirements Management. Meist waren schon zu Beginn die Anforderungen nicht ausreichend geklärt und damit konnte auf deren Änderungen auch nicht richtig reagiert werden. Das Buch bietet einen Überblick über Theorie und Praxis des Requirements Management. Es beschreibt, wie Anforderungen entwickelt, gesammelt, dokumentiert und im Projekt verfolgt werden. Die grundsätzlichen Methoden, Verfahren, Werkzeuge und Notationen des Requirements Management werden übersichtlich behandelt. Sie werden durch konkrete Beispiele aus der Projektarbeit illustriert.

Als Beispiel einer modernen Methode der Anforderungsbeschreibung werden Use-Case-Szenarien in der UML-Notation verwendet. Praktische Fallstudien unterstützen die konkrete Umsetzung.

Leser: Produktmanager, Projektleiter, Softwareentwickler

Weitere Informationen finden Sie unter

`http://www.dpunkt.de/buch/3-89864-336-0.html`

Sneed, H.M.:

Software-Projektkalkulation – Wissen was Projekte wirklich kosten

Hanser-Verlag, 2005 (228 Seiten)
ISBN 3-446-40005-2

Wer einmal die Kosten oder die Zeit für ein Software-Projekt falsch kalkuliert hat, weiß, dass kein Unternehmen sich das öfter leisten kann. Projektkalkulation ist eine Überlebensfrage der Software-Industrie. Für Auftragnehmer wie für Auftraggeber ist die richtige Kalkulation unabdingbar für den Projekterfolg.

Die meisten Techniken für Aufwandsschätzung, die in der Praxis verbreitet sind, eignen sich nur bei IT-Projekten für eine Neuentwicklung. Geht es in Ihrem Projekt jedoch um Wartung, Migration, Integration oder Sanierung, so müssen Sie darauf abgestimmte Methoden einsetzen. Dieses Buch zeigt, welche Techniken der Aufwandsschätzung für welche Art von Projekten zu nutzen sind.

Deek, F.P.; McHugh, J.A.M.; Eljabiri, O.M.:

Strategic Software Engineering

Auerbach Publications, 2005 (333 pages)
ISBN 0-8493-3939-1

Strategic Software Engineering: An Interdisciplinary Approach presents software engineering as a strategic, business-oriented, interdisciplinary endeavour, rather than simply a technical process, as it has been described in previous publications.

The book addresses technical, scientific, and management aspects of software development in a way that is accessible to a wide audience. It provides a detailed, critical review of software development models and processes, followed with a strategic assessment of how process models evolved over time and how to improve them. The authors then focus on the relation between problem-solving techniques and strategies for effectively confronting real-world business problems. They also analyze the impact of interdisciplinary factors on software development, including the role of people and business economics. The book concludes with a brief look at specialized system development.

The diverse backgrounds of the authors, encompassing computer science, information systems, technology, and business management, help create this book's integrated approach, which answers the demand for a comprehensive, interdisciplinary outlook that covers all facets of how software relates to an organization.

Contents:

Provides a detailed, critical review of software development models and processes, introduces and critiques the basic software development process and key risk-reduction models, Examines the theme of process improvement and explores recent trends in software process models, relates classic problem-solving concepts to software development and addresses how software tools influence problem-solving, explains how the focus of development has shifted from technical to business contexts, discusses people-related drivers for development, focuses on the role of costs and economics in software engineering.

El Emam, K.:

The ROI from Software Quality

Auerbach Publications, 2005 (279 pages)
ISBN 0-8493-3298-2

The ROI from Software Quality provides the tools needed for software engineers and project managers to calculate how much they should invest in quality, what benefits the investment will reap, and just how quickly those benefits will be realized. This text provides the quantitative models necessary for making real and reasonable calculations and shows how to perform ROI analysis before and after implementing a quality program. The book demonstrates how to collect the appropriate data and easily perform the appropriate ROI analysis.

Taking an evidence-based approach, this book supports its methodology with large amounts of data and backs up its positioning with numerous case studies and straightforward return-on-investment calculations. By carefully substantiating arguments numerically, this volume separates itself from other works on ROI.

Contents:

Explores options that allow managers to prioritize in pursuit of quality, enables quality benchmarking by including referenced examples of quality practices and implementations, explains in detail how to justify ROI calculations, delivers concrete data on the benefits of specific software engineering practices, provides a comprehensive analysis of the quality and security of open source software (OSS), includes implementation guidelines for using ROI within a software development organization, contains more than 200 tables for easy reference.

Abran, A.; Bundschuh, M.; , Büren, G.; Dumke, R. (Eds.):

Software Measurement – Research and Application

Springer Publ., Aachen, 2004 (602 pages)
ISBN 3-8322-3383-0

This proceedings of the joined conferences, the 14th International Workshop on Software Measurement (IWSM 2004) and the DASMA MetriKon 2004, try to reflect a bit of all the concepts developed and the experiences made when measuring software. They are of particular interest to software engineering researchers, as well as to practitioners, in the areas of project management and quality improvement programs, for both software maintenance and software development.

Ebert, C.; Dumke, R.; Bundschuh, M.; Schmietendorf, A.:

Best Practices in Software Measurement

Springer Publ., 2004 (320 pages)
ISBN 3-540-20867-4

The software business is challenging enough without having to contend with recurring errors. One way repeating errors can be avoided is through effective software measurement. In this book is offered a practical guidance built upon insight and experience. The authors detail knowledge and experiences about software measurement in an easily understood, hands-on presentation and explain many current ISO standards (see also **`http://metrics.cs.uni-magdeburg.de/`**).

Chrissis, M.B.; Konrad, M.; Shrum, S.:

CMMI – Guidelines for Process Integration and Product Improvement

Addison-Wesley, 2004 (663 pages)
ISBN 0-321-15496-7

This book is the definitive reference for the most current release of CMMI models. To use a CMMI model available on the SEI Web site, users must choose from among multiple models based on their organization's improvement needs. This book provides a single source for all CMMI model information. Readers can get started without having to select a model first – all of the choices are compiled in one place and explained in detail.

The book begins with background information needed to understand the content and structure of these integrated models and how to use them. A case study illustrates their implementation in a real environment. A variety of practical material, such as glossary and index, is also provided. The bulk of the book comprises the content of all CMMI models, covering the 25 process areas (PAs) that span the product life cycle, including detailed best practices.

Preprints/Technical Reports:

Dumke, R.; Schmietendorf, A.; Zuse, H.: *Formal Description of Software Measurement and Evaluation*. University of Magdeburg, 2005

Braungarten, R.; Kunz, M.; Dumke, R.: *An Approach to Classify Software Measurement Storage Facilities*. University of Magdeburg, 2005

see as pdf files:

`http://ivs.cs.uni-magdeburg.de/sw-eng/agruppe/forschung/Preprints.shtml`

IASTED SE 2006:

***IASTED International Conference on Software Engineering* 2006**
February 14-16, 2006, Innsbruck, Austria
see: **http://www.iasted.org/conferences/2006/innsbruck/se.htm**

SEPG 2006:

18th Software Engineering Process Group Conference
March 6-9, 2006, Nashville, Tennessee
see: **http://www.sei.cmu.edu/sepg/index.html**

SMEF 2006:

Software Measurement European Forum
May 10-12, 2006, Rome, Italy
see: **http://www.iir-italy.it/smef2006/**

CSMR 2006:

10th European Conference on Software Maintenance and Reengineering
March 22-24, 2006, Bari, Italy
see: **http://serlab.di.uniba.it/csmr2006/**

EASE 2006:

10th International Conference on Empirical Assessment in Software Engineering
April 10-11, 2006, Staffordshire, UK
see: **http://ease.cs.keele.ac.uk/**

SPICE 2006:

6th International SPICE Conference on Process Assessment and Improvement
May 3-5, 2006, Luxembourg
see: **http://www.ifi.uni-klu.ac.at/Conferences/SPICE2005_**

PQST 2006:

International Conference on Practical Software Quality & Testing
May 1-5, 2006, Las Vegas
see: **http://www.psqtconference.com/2006west/_**

WWW 2006:

15th International World Wide Web Conference
May 23-26, 2006, Edinburgh, UK
see: **http://www2006.org/_**

IWPC 2006:

14th International Workshop on Program Comprehension
June 14-16, 2006, Athens, Greece
see: `http://www.icpc2006.uwaterloo.ca/`

PROMISE 2006:

2nd International Workshop on Predictor Models in Software Engineering
September 24, 2006, Philadelphia, USA
see: `http://promise.unbox.org/Promise2006`

ICSE 2006:

International Conference on Software Engineering
May 20-28, 2006, Shanghai, China
see: `http://www.isr.uci.edu/icse-06/`

SIGMetrics 2006:

ACM SIGMetrics - Performance 2006
June 26-30, 2006, Saint-Malo, France
see: `http://www.cs.wm.edu/sigm06/`

ESEPG 2006:

11th European Software Engineering Process Group Conference
June 12-15, 2006, Amsterdam, Netherlands
see: `http://www.espi.org/sepg/_`

PROFES 2006:

6th International Conference on Product Focused Software Process Improvement
June 12-14, 2006, Amsterdam, Netherlands
see: `http://www.cwi.nl/events/2006/profes/`

QATWBA 2006:

2nd International Workshop on Quality Assurance and Testing of Web-Based Applications
September 18-21, 2006, Chicago, USA
see:
`http://conferences.computer.org/compsac/2006/QATWBA2006CFP.htm`

IWSM 2006:

16th International Workshop on Software Measurement
November 2-3, Potsdam, Germany
see: `http://iwsm2006.cs.uni-magdeburg.de`

QEST 2006:

3rd International Conference on Quantitative Evaluation of SysTems
September 11-14, 2006, Riverside, California, USA

see: **http://www.qest.org/**

UKSMA 2006:

17th Annual UKSMA Conference - Managing your Software (through Measurement)
October 11-12, 2006, London, UK
see: **http://www.uksma.co.uk/**

ISESE 2006:

ACM-IEEE 5th International Symposium on Empirical Software Engineering
September 21-22, 2006, Rio de Janeiro, Brazil
see: **http://www.cos.ufrj.br/~ght/isese2006**

QFD 2006:

18th Symposium on Quality Function Deployment
December 2, 2006, Austin, Texas
see: **http://www.qfdi.org/call_for_papers.htm**

see also: **OOIS, ECOOP** and **ESEC** European Conferences

Other Information Sources and Related Topics

- **`http://rbse.jsc.nasa.gov/virt-lib/soft-eng.html`**
  Software Engineering Virtual Library in Houston

- **`http://www.mccabe.com/`**
  McCabe & Associates. Commercial site offering products and services for software developers (i. e. Y2K, Testing or Quality Assurance)

- **`http://www.sei.cmu.edu/`**
  Software Engineering Institute of the U. S. Department of Defence at Carnegie Mellon University. Main objective of the Institute is to identify and promote successful software development practices.
  Exhaustive list of publications available for download.

- **`http://dxsting.cern.ch/sting/sting.html`**
  Software Technology Interest Group at CERN: their WEB-service is currently limited (due to "various reconfigurations") to a list of links to other information sources.

- **`http://www.spr.com/index.htm`**
  Software Productivity Research, Capers Jones. A commercial site offering products and services mainly for software estimation and planning.

- **`http://www.qucis.queensu.ca/Software-Engineering/`**
  This site hosts the World-Wide Web archives for the USENET usegroup comp.software-eng. Some links to other information sources are also provided.

- **`http://www.esi.es/`**
  The European Software Institute, Spain

- **`http://www.lrgl.uqam.ca/`**
  Software Engineering Management Research Laboratory at the University of Quebec, Montreal. Site offers research reports for download. One key focus area is the analysis and extension of the Function Point method.

- **`http://www.SoftwareMetrics.com/`**
  Homepage of Longstreet Consulting. Offers products and services and some general information on Function Point Analysis.

- **`http://www.utexas.edu/coe/sqi/`**

Software Quality Institute of the University of Texas at Austin. Offers comprehensive general information sources on software quality issues.

- **http://wwwtrese.cs.utwente.nl/~vdberg/thesis.htm**
  Klaas van den Berg: Software Measurement and Functional Programming (PhD thesis)

- **http://divcom.otago.ac.nz:800/com/infosci/smrl/home.htm**
  The Software Metrics Research Laboratory at the University of Otago (New Zealand).

- **http://ivs.cs.uni-magdeburg.de/sw-eng/us/**
  Homepage of the Software Measurement Laboratory at the University of Magdeburg.

- **http://www.cs.tu-berlin.de/~zuse/**
  Homepage of Dr. Horst Zuse

- **http://dec.bournemouth.ac.uk/ESERG/bibliography.html**
  Annotaded bibliography on Object-Oriented Metrics

- **http://www.iso.ch/9000e/forum.html**
  The ISO 9000 Forum aims to facilitate communication between newcomers to Quality Management and those who have already made the journey have experience to draw on and advice to share.

- **http://www.qa-inc.com/**
  Quality America, Inc's Home Page offers tools and services for quality improvement. Some articles for download are available.

- **http://www.quality.org/qc/**
  Exhaustive set of online quality resources, not limited to software quality issues

- **http://freedom.larc.nasa.gov/spqr/spqr.html**
  Software Productivity, Quality, and Reliability N-Team

- **http://www.qsm.com/**
  Homepage of the Quantitative Software Management (QSM) in the Netherlands

- **http://www.iese.fhg.de/**
  Homepage of the Fraunhofer Institute for Experimental Software Engineering (IESE) in Kaiserslautern, Germany
- **http://www.highq.be/quality/besma.htm**
  Homepage of the Belgian Software Metrics Association (BeSMA) in Keebergen, Belgium

- **http://www.cetus-links.org/oo_metrics.html**
  Homepage of Manfred Schneider on Objects and Components

- **http://dec.bournemouth.ac.uk/ESERG/bibliography.html**
  An annotated bibliography of object-oriented metrics of the Empirical Software Engineering Research Group (ESERG) of the Bournemouth University, UK

News Groups

- **news:comp.software-eng**
- **news:comp.software.testing**
- **news:comp.software.measurement**

Software Measurement Associations

- **http://www.dasma.org**
  DASMA Deutsche Anwendergruppe für SW Metrik und Aufwands-schätzung e.V.

- **http://www.aemes.fi.upm.es**
  AEMES Association Espanola de Metricas del Software

- **http://www.cosmicon.com**
  COSMIC Common Software Measurement International Consortium

- **http://www.esi.es**
  ESI European Software Engineering Institute in Bilbao, Spain

- **http://www.mai-net.org/**
  Network (MAIN) Metrics Associations International

- **http://www.sttf.fi**
  FiSMA Finnish Software Metrics Association

- **http://www.iese.fhg.de**
  IESE Fraunhofer Einrichtung für Experimentelles Software Engineering
- **http://www.isbsg.org.au**
  ISBSG International Software Benchmarking Standards Group, Australia

- **http://www.nesma.nl**
  NESMA Netherlands Software Metrics Association

- **http://www.sei.cmu.edu/**
  SEI Software Engineering Institute Pittsburgh

- **http://www.spr.com/**
  SPR Software Productivity Research by Capers Jones

- **http://fdd.gsfc.nasa.gov/seltext.html**
  SEL Software Engineering Laboratory - NASA-Homepage

- **http://www.vrz.net/stev**
  STEV Vereinigung für Software-Qualitätsmanagement Österreichs

- **http://www.sqs.de**
  SQS Gesellschaft für Software-Qualitätssicherung, Germany

- **http://www.ti.kviv.be**
  TI/KVIV Belgish Genootschap voor Software Metrics

- **http://www.uksma.co.uk**
  UKSMA United Kingdom Software Metrics Association

Software Metrics Tools (Overviews and Vendors)

Tool Listings

- **http://www.cs.umd.edu/users/cml/resources/cmetrics/**
  C/C++ Metrics Tools by Christopher Lott

- **http://mdmetric.com/**
  Maryland Metrics Tools

- **http://cutter.com/itgroup/reports/function.html**
  Function Point Tools by Carol Dekkers

- **http://user.cs.tu-berlin.de/~fetcke/measurement/products.html**
  Tool overview by Thomas Fetcke

- **http://zing.ncsl.nist.gov/WebTools/tech.html**
  An Overview about Web Metrics Tools

Tool Vendors

- **http://www.mccabe.com**
  McCabe & Associates

- **http://www.scitools.com**
  Scientific Toolworks Inc.

- **http://zing.ncsl.nist.gov/webmet/**
  Web Metrics

- **http://www.globalintegrity.com/csheets/metself.html**
  Global Integrity

- **http://www.spr.com/**
  Software Productivity Research (SPR)

- **http://jmetric.it.swin.edu.au/products/jmetric/**
  JMetric

- **http://www.imagix.com/products/metrics.html**
  Imagix Power Software

- **http://www.verilogusa.com/home.htm**
  VERILOG (LOGISCOPE)

- **http://www.qsm.com/**
  QSM

METRICS NEWS

VOLUME 10 **2005** **NUMBER 2**

CONTENTS

ISSN 1431-8008