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

Editors:

ALAN ABRAN
Professor and Director of the Research Lab. in Software Engineering Management
Quebec-University of Montreal
Departement of Computer Science
C.P. 8888 Succursale Centre-Ville, Montreal, H3C 3P8, Canada
Tel.: *+1-514-987-3000, -89000,* ***Fax:*** *+1-514-987-8477*
Email: *abran.alain@uqam.ca*

MANFRED BUNDSCHUH
Chair of the DASMA
Sander Höhe 5, 51465 Bergisch Gladbach, Germany
Tel.: *+49-2202-35719*
Email: *Bundschuhm@acm.org*
http://www.dasma.de

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*
Email: *dumke@ivs.cs.uni-magdeburg.de*

CHRISTOF EBERT
Dr.-Ing. in Computer Science
Alcatel Telecom, Switching Systems Division
Fr. Wellensplein 1, B-2018 Antwerpen, Belgium
Tel.: *+32-3-240-4081,* ***Fax:*** *+32-3-240-9935*
Email: *christof.ebert@alcatel.de*

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*
Email: *zuse@tubvm.cs.tu-berlin.de*

Editorial Office: Otto-von-Guericke-University of Magdeburg, FIN/IVS, Postfach 4120, 39016 Magdeburg, Germany
Technical Editor: DI Erik Foltin
The journal is published in one volume per year consisting of two numbers.

W o r k s h o p P r o g r a m

of the 10th International Workshop on Software Measurement
in Berlin, Germany, Oct. 4-6, 2000

WEDNESDAY, October 4, 2000:

10.00 - 10.15 Welcome and Introduction by A. Abran and R. Dumke

Chair: *R. Dumke, University of Magdeburg*

10.15 - 10.45 RHEINDT, M., Siemens AG, Germany:
How to Start with an Optimal Metrics System

10.45 - 11.15 BUGLIONE et al., SBU Telecoms, Italy:
QF2D: A different way to measure Software Quality

11.15 - 11.30 Refreshments

Chair: *H. Sneed, SES Munich*

11.30 - 12.00 BEYER et al., University of Cottbus, Germany:
Impact of Inheritance on Metrics for Size, Coupling, and Cohesion in Object-Oriented Systems

12.00 - 12.30 XINKE, Li et al., Hefei University, China:
A Measurement Tool for Object Oriented Software and Measure Experiments with it Measurements on the ERP Requirements

12.30 - 14.00 Lunch

Chair: *H. Zuse, Technical University of Berlin*

14.00 - 14.30 SANTILLO, L., Italy:
Early & Quick COSMIC-FFP Analysis using Analytic Hierarchy Process

14.30 - 15.00 SNEED, H., SES, Germany:
Extraction of Function-Points from Source Code

15.00 - 15.30 FOLTIN et al., University of Magdeburg, Germany:
Estimation the Cost of Carrying out Tasks Relating to Performance Engineering

15.30 - 15.45 Refreshments

Chair: *A. Abran, Universite du Quebec, Montreal*

15.45 - 16.15 DANEVA, M., Clearnet Comm., Canada:
An Assessment of the Effects of Requirements Reuse Measurement on the ERP Requirements Engineering Process

16.15 - 16.45 DUMKE et al., University of Magdeburg, Germany:
A New Metrics-Based Approach for the Evaluation of Customer Satisfaction in the IT Area

19.00 Visit of the Museum of Telecommunication

THURSDAY, October 5, 2000:

Chair: *C. Lewerentz, Technical University of Cottbus*

9.00 - 9.30 SATPATHY,M. et al., University of Reading, UK:
A Generic Model for Assessing Process Quality

9.30 - 10.00 HAMANN, D. et al., IESE, Germany:
Using FAME Assessments to Define Measurement Goals

10.00 - 10.15 Refreshments

Chair: *T. Hall, University of Hertfordshire*

10.15 - 10.45 SARFERAZ et al., University of Marburg, Germany:
CEOS - a Cost Estimation Method for Evolutionary, Object-oriented Software Development

10.45 - 11.15 SCHMELZ et al., University of Jena, Germany:
Utility Metrics for Economic Software Agents

11.15 - 11.45 EBERT, C., Alcatel, Belgium:
Experiences with Validation Activities in a Global Development

11.45 - 13.30 Lunch

Chair: *C. Ebert, Alcatel Belgium*

13.30 - 14.00 HALL, T. et al., University of Hertfordshire, UK:
Measurement in Software Process Improvement Programmes: an Empirical Study

14.00 - 14.30 SCHMIETENDORF et al., Deutsche Telekom, Germany:
Maturity Evaluation of the Performance Engineering Process

14.30 - 14.45 **Refreshments**

14.45 - 16.30 *Constituent assembly of the GI FG 2.1.10/*
Sight Seeing of Berlin for non GI members

19.00 **Social event**

FRIDAY, October 6, 2000:

Chair: *R. v. Solingen, IESE Kaiserslautern*

9.00 - 9.30 BLACK et al., South Bank University, UK:
Measuring the Ripple Effect of Pascal Programs

9.30 - 10.00 DION et al., CGL Inc., Canada:
Mapping Processes Between Parallel, Hierarchical and Orthogonal System Representations

10.00 - 10.15 **Refreshments**

Chair: *M. Bundschuh, DASMA Bonn*

10.15 - 10.45 ESTOL, C., University of Belgrano, Argentina:
Developing an IT Sorecoard for IT Departments: The Use of IT Indicators

10.45 - 11.15 ABRAN et al., University du Quebec, Montreal:
A COSMIC Experiment Report

11.15 - 12.30 **Final discussion**

Hotel Reservation:

Bildungs- und Tagungsstätte Berlin
Berliner Strasse 16 A
D-15711 Königs Wusterhausen
Germany

Tel.: +49 (3375) 249 - 100

Workshop Registration (until September 30, 2000)

University of Magdeburg
Dept. of Computer Science
Postfach 4120
D-39016 Magdeburg
Germany

Dagmar Dörge

Tel.: +49 (391) 67 - 18664
Fax: +49 (391) 67 - 12810
Email: doerge@ivs.cs.uni-magdeburg.de

DASMA e.V.

The DASMA Fall Conference 2000 will be held at the

November 30 to ***December 1***, ***2000***,

in ***Düsseldorf*** at ***Fa. Itergo***.

Some of the lectures in the programme are

- **Martin Hooft von Huisduynen:**
  Does E-Commerce really provide the results we expect from it and how do we know?

- **Rini van Solingen:**
  Practical Use of Software Metrics

- **Reiner Dumke:**
  How many software metrics are necessary in the IT area?

Workshop Report

1. German Workshop on Performance Engineering within the Software Development

Andreas Schmietendorf, André Scholz

Introduction

The "First German Workshop on Performance Engineering within the Software Development", which was co-organized with the "German Association of Computer Science", took place in Darmstadt on the 15th of July 2000.

Mr. Seemke, who is the managing director of DeTeCSM Technical Support, inaugurated the workshop and underlined the increasing importance of performance engineering. Increasing customer requirements on the proofing of service levels from the business perspective make it necessary to work out rules and techniques for a performance-oriented system development.

Intentions

One of the most critical non-functional quality factors is the performance characteristic of a software system. Performance can be defined as the ability of a system to process a given amount of tasks in a determined time interval. Thus, performance stands in a direct relation to the speed of accomplishing business processes. Following a research study of Gartner Group, the competition between individual companies will be mainly influenced by the efficiency of the business processes. It is the duty of the information management to integrate a company's business processes into the process network of the customers and subcontractors with regard to performance aspects.

The early consideration of performance characteristics of an information system is the main idea of performance engineering. Thus, systems can be built, which fulfil defined resource consumption as well as defined performance characteristics. In practice, active performance analyses are often neglected. This quality factor is often only considered at the end of the software development process. At that stage performance problems lead to expensive tuning measures, to investments into new hardware or to a redesign of the software application.

Although the performance of hardware system still increases, complex application systems on the basis of new hardware and software system require an explicit consideration of the performance characteristics within all development phases.

Program Overview

The workshop was distinguished into 3 sessions and a final discussion:

- PROCESSES, ***Chair:*** ***Prof. Dr. F. Lehmann (University of the German Armed Forces, Munich)***
- METHODS, ***Chair:*** ***Prof. Dr. R. Dumke (University of Magdeburg)***
- BENCHMARKING / MONITORING, ***Chair:*** ***Prof. Dr. C. Rautenstrauch (University of Magdeburg)***

In total, nine papers focussing on relevant topics of performance engineering were presented:

***R. Dumke (University of Magdeburg)**, Conception of a web-based SPE Development Infrastructure:*
The presentation analyzed existing models and methods for a performance-oriented system development. It was based on a general Measurement Framework, which was developed at the University of Magdeburg. The World Wide Web was integrated into the development process as an information source.

***R. Gerlich (BSSE)**, Built-in Performance and Robustness Engineering Capabilities by a Formalized and automated Software Development Process:*
The paper proposed an automated software development process on the basis of a formal specification. It combined the application generation with verification and validation methods. Thus, functional and non-functional properties, e.g. performance and robustness, could be analyzed.

***A. Scholz, A. Schmietendorf (University of Magdeburg)**, Performance Engineering – Duties and Responsibilities:*
It was the aim of this presentation to give a comprehensive introduction into the complex field of performance engineering. Therefore, aims, useable methods and models were discussed. Furthermore, performance engineering was differentiated of related tasks and fields.

***E. Dimitrov (T-Nova Research Center Berlin)**, UML-based Performance Engineering:*
Today, UML is often used for object-oriented modeling. This presentation analyzed the possibilities for performance modeling with the help of UML. Therefore, existing proposals were evaluated and modeling requirements were summarized.

***L. Kerber (University of Erlangen-Nürnberg)**, Scenario-based Performance Evaluation of SDL/MSC- Systems:*
The application of Performance Message Sequence Charts (PMSC) was presented in the field of protocol design. PMSC offer a model for early performance predictions. Furthermore, time-related requirements on systems can be specified.

***A. Kraiß (Dresdner Bank)**, Target-oriented Performance Engineering on the basis of Message-oriented Middleware:*
The paper proposed a model for an automated task prioritization at a server when dealing with predefined response times and different user categories. A tool was sketched on the basis of the results of stochastical queuing models. The tool was in the position to outline the performance limits of a specific architecture and to adjust the priorities of the tasks dynamically to assure defined response times.

***R. Koeppe (University of Magdeburg)**, Bus Protocol Design under Performance Aspects:*
The paper described a new solution principle for an exact determination of guaranteed response times. The implementation was derived from the protocol specification. With the help of this principle, the performance of different design alternatives can be already compared at early development stages.

***H. Ultsch (Zott + Co)**, Performance Analysis with the Workload Driver s_aturn referring to DIN 66273 / ISO 14756:*

DIN 66273 specifies an evaluation process for the performance characteristics of an IT-system from the user point of view. A black box test driver simulated a given amount of users. The amount of users, time and process characteristics could be adjusted to real values.

***K. Wilhelm (University of Essen)**, The Use of Monitoring and Benchmarking within the R/3 Performance Engineering:*
This paper described a measurement concept for monitoring SAP R/3 systems. Therefore, the measurement tool "R/3-Live Monitor" was introduced. The importance of benchmarking and monitoring was explained using concrete examples.

Results

Following assessments of the participants, the workshop was very successfully. Especially, the inter-disciplinary discussion of performance engineering (software engineering, performance modeling and business information systems) seemed to make the deeper consideration within the industry as well as within the curricular of the universities possible.

The participants decided to set up a working group to develop this field of research further. The working group is primary assigned to the “GI-Fachgruppe 3.2.1 Measurement, Modeling and Evaluation of Computer Systems”. Furthermore, the working group is supported by the “GI-Fachgruppe 2.1.10 Software Measurement and Evaluation as well as by the “Fachbereich 5 Business Information Systems”. Further on, a cooperation with the working groups within the USA as well as within the UK is aimed.

Speaker of the Working Group:

Andreas Schmietendorf
T-Nova
Research Center Berlin,
Wittestr. 30H, 13509 Berlin
Germany

Email: A.Schmietendorf@telekom.de

André Scholz
University of Magdeburg
Business Information System Research Group,
Universitätsplatz 2, 39106 Magdeburg
Germany
Email: ascholz@iti.cs.uni-magdeburg.de

Further information is available on the following web-side:

http://www-wi.cs.uni-magdeburg.de/pe2000/

The program and organization committee would like to take this opportunity and thank the benchmark lab in Darmstadt for providing excellent working conditions.

Possibilities of the Description and Evaluation of Software Components

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

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

* *T-Nova Deutsche Telekom Innovationsgesellschaft mbH, Entwicklungszentrum Berlin, Wittestraße 30 N, 13509 Berlin, E-Mail: A.Schmietendorf@telekom.de*

ABSTRACT

The re-use of software components during the software development is considered to be an important factor to improve the quality and productivity and thus to reduce the time to market of the final product. In this paper we will present a proposal for a description model for re-usable components. We will also present the results of case studies concerned with both telecom specific and „generic" IT-components. These components have been examined using the description model and a further set of (empirical) criterions. Based on the results a model concept for the empirical assessment of JavaBeans, which is currently under development, is presented.

1 Introduction

Within the industrial production of goods there is a high degree of using pre-constructed intermediate products. The intention is to increase the productivity, the quality of the final products and to reduce the time to market. Basis for such a production run is a shared process between the "suppliers" of intermediate products and the "assembly" of the final product. Often standards are used to define and assess the quality and the functionality of the intermediate products to assure the fulfilment of the customer requirements for the final product. Based on the experience within the "traditional" industry, re-use is considered to be a promising attempt to transform the manufacturing approach, which is still typical for a wide range of today's software development projects, into an engineering discipline.

Based on the concept of inheritance, object oriented software development is expected to allow for a high degree of re-use. However, as early as 1994 the hypothesis was formulated by Udell (in [15]), that object oriented software development will not meet the expectation of high re-use rates. He suggested that software components may be better suited for re-use. Crucial issues for using components is how to identify them and how to describe their qualitative and functional properties to allow an easy identification, retrieval and finally re-use of the respective component. In the following we will therefore examine ideas for component description models and assess existing software development projects which use components to some degree. Furthermore we will discuss hypothesises on the connection of software quality and re-use. Based on this and on empirical investigations using Java Beans we will present a concept to obtain valid metrics to prove or falsify the above hypothesises. The ultimate goal is to derive a set of metrics that allows an objective quality assessment of components.

The following is based on the research project SW-WiVe[1] carried out at the Deutsche Telekom AG Entwicklungszentrum Berlin in co-operation with the University of Magdeburg. Besides the results presented in this paper, the project was focused on the identification and definition of the necessary re-use procedures and required resources. It also included a set of metrics for the assessment and controlling of the re-use process as well as a suggestion of a model to assess the maturity of re-use on an organisational level. Details of these results are given in [12].

2 Description of components

2.1 Existing description forms

Quite a few approaches to describe re-usable components (assets) can be found in literature. A very broad description is given in [4]: an asset is characterised by a "description" and a "body" as shown in
Figure **1**. The description contains basically meta-information aimed at describing and managing the component. The body is the actual "work-product". Typically the body does not contain just one "work-product" but a set thereof. Each work-product represents a different abstraction of the same peace of software (i. e. conception, analysis, design, implementation or test models). Based on the model given in [4] a minimal required description for assets is given at (http://direct.asset.com). Especially developers often need a more detailed description of the components they are handling. In practice, different forms of descriptions are in use for different components.

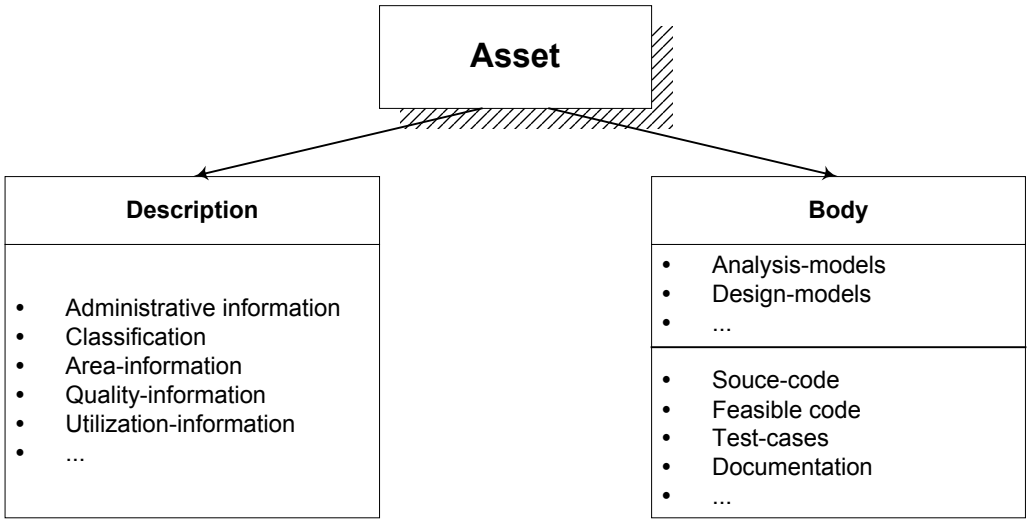

Figure 1: Asset Model (in [4])

In the following we will focus on the special description forms for pattern (a sub-set of assets), which are already in use. A Pattern is not tied to a specific form, rather there are a lot of possible schemata to describe patterns. Which form should be chosen depends on the viewpoints of the intended users of the pattern description: is just an overview of the basic idea of the component required or is more detailed information required to decide whether the pattern solves a specific requirement:

[1] WiVe: acronym for "Wiederverwendung", the German word for re-use

- For **project managers** (or managers in general) often a short description of the basic idea of the pattern is best suited.
- **developers** need a more detailed pattern description. This description should contain information on the different roles to be implemented, a description on the co-operation of the different roles, a development strategy and further aspects which should be considered in the context of the development.

The description and the representation of components is also depended upon the type of component. For analysis and design pattern a graphical representation is useful and often sufficient. Implementation pattern however need more detailed descriptions in textual or pseudo code form. The following should apply in general for all pattern descriptions:

- it describes a solution of a recurring problem,
- this solution has been used in practice,
- the solution (or its general idea) is applicable to a certain range of requirements or can be adopted,
- the solution contains a description of the actual design structure solving the problem as well as a description of a process to derive this structure.

A general pattern description template contains details on:

- Context: description of the environment constituting the problem
- Problem:
  - Requirements for the solution
  - Constraints that have to be considered
  - Desirable properties of the solution
- Solution: description of the problem solving idea.

Example description forms are the GoF-Form [7], the POSA-Form [2], the Coplien-Form [10] and the Portland-Form (Ward Cunningham [10]). The GoF-Form (Gang of Four), a detailed description template focused mainly on functional issues, seems to be the most widely acknowledged among practitioners. Other description forms (e. g. in [11]) focus more on qualitative aspects.

2.2 Proposal for component description for Deutsche Telekom AG

The following table presents the component description issues that have been identified by the project SW-WiVe. This description template forms the basis of a virtual re-use repository. This repository will be used in combination with a metrics database.

Aspect	Issues
Identification	– Component Name – Unique Identificator – Alias

Aspect	Issues
Administrative Information (History)	– Supply Date – Release Date – Change Reason – Author – Producer / Company – Licensing
Domain Specific Details	– Problem area – Context – References – Standards
Technical Details	– Format – Language – Component Size – Architecture of the Component – Interfaces – Sample Solution
Quality (ISO 9126)	– Portability – Usability – Efficiency – Functionality – Reliability – Maintainability
Access	– Access type (HTTP, FTP, Repository,...) – Access Frequency – Usage Frequency

Table 1: Description template for components

3 CASE STUDIES

3.1 Intention of the case studies

During the course of the project "SW-WiVe" some of the components that are already used within the Deutsche Telekom AG had been examined. The asset description template presented in 2.2 was used. We also applied an additional catalogue of additional criterions to assess the usability of components.

The goal was to derive conclusions concerning the integration of component re-use within the development process and its impact on quality and productivity of the final product. Another result of the case studies was the identification of weak spots in the current re-use attempts and to formulate a basis for a metrics-based re-use process.

3.2 Components for Intelligent Network Applications

The Intelligent Network (IN) is as well a network architecture as a technology to develop and to provide telecommunication services. The IN exists in addition to the standard telecommunication network and supports a certain control function. It's aim is to process

connect requests (Calls) using a simple processing logic. For the development of IN-services (e. g. the Televotum service TED) “Service Independent building Blocks” (SIBs) are used. SIBs are re-usable components, each describing a complex activity. The composition of several SIBs yields a certain IN-service. Based on this modular concept it is possible to delegate similar tasks of distinct services to one SIB and to re-use the functionality of existing SIBs for future development.

The following are the characteristic features of SIBs:

- SIBs are used to model services,
- SIBs are standardised, re-usable and transparent entities,
- SIBs exist independent of the specific service they describe,
- SIBs are independent of architecture consideration,
- SIBs use stable, standardised interfaces, each consisting of one or more inputs and one or more outputs.

New IN services can be developed using graphic oriented development environments by combining re-usable SIBs without changing the whole IN system. All information concerned with the data environment and a few additional information belong to the description of a SIB. The following text-based description template is used to describe the SIB properties:

Mandatory:
- Textual definition of the SIB,
- most SIBs need a specific **input parameter**, e. g. search keys in SIB ´getUserRec´ or a default announce identification in SIB ´Announce´.
- Service Support Data (SSD) are static data and tables available and needed throughout the SIB’s execution, e. g. tariff data,
- call Instance Data (CID) are dynamic data available only during a specific call, they are deleted after the call has been completed. They are divided into input and output CID,
- Functions – Description of the operations carried out by the SIB.

Additional Information:
- Identification of potential services that may use the SIB,
- Graphical representation,
- Standard Description Language (SDL) diagrams.

A formalised and compressed language description (similar to IDL concepts) is often sufficient for designers and developers of IN-services. Such a description templates contains usually:

- (extended) textual definition,
- Input parameter,
- Output parameter,
- Description of attributes.

3.3 Components for TMN-Applications

The development of information systems for managing public telecommunication networks (TMN Telecommunication-Management-Network) is done using so-called Managed Objects

(MO). A MO is a re-usable component, it is an abstract representation of a resource of the telecommunication system (e. g. a switching device) in a form meaningful for the information system. The properties of a Managed Object are defined as follows:

- **Attributes and Attribute Groups**, providing the internal states for external evaluation or modification operations,
- **Actions**, can be performed by the MO,
- **Behaviour** as a reaction on events or as a result of performing an operation,
- **Notifications** can be send by the MO,
- **Parameter** controlling the execution of the actions,
- **Name Binding** to realise a hierarchy: MOs are identified by their name and are organised by the MO-tree in the Management Information Base (MIB),
- **Packages** containing attributes, attribute groups, messages and operations.

A system independent specification of data and protocols is achieved using formal specification languages. ASN.1 (Abstract Syntax Notation 1, X.208) and a derived language GDMO (Guidelines for the definition of Managed Objects, X.722) have been developed for this purpose. Basic ideas of the specification is the abstraction of the properties of a resource to be managed and using this abstraction for the managing system, the modularization of management information using packages and an object oriented design of the “Managed Object Classes” using the well-known concepts of inheritance and encapsulation. The data declaration of Managed Objects with the same properties is called a “Managed Object Class”.

3.4 Components based on Java Beans

The platform independent and distributable Java Beans (which can also co-operate with non native Java applications) may be a possible platform for a component based software development. The JavaBean-Model from SUN Microsystems allows to implement components using Java. The programming language Java has not been altered for this purpose, but the JDK[2] has been extended by one more package. This package contains all classes, interfaces and exceptions needed to develop JavaBeans.

Java Beans are, according to the specification by SUN, re-usable software components. Their properties may be changed using visual builder tools to reflect changes and adaptations in the system requirements. By combining several beans it is possible to develop "classical" applications as well as Java Applets. It is also possible to create new beans using existing ones. There is a distinction between beans with a graphical (user visible) representation and "invisible" beans which are used to represent data base access or resources in general.

The following items explain the core principles of Java bean technology:

Introspection: This (implicit and explicit) property allows to request information on the functions and properties (attributes) of the bean during run-time. Especially information on the methods and events declared as "public" can be requested. Builder tools using the introspection property therefore do not need to have access to the source code. The implicit Introspection function relies on a simple **Reflection-Mechanism**, which returns all methods provided by the bean. **Design Patterns** are used to draw conclusions on further properties,

[2] JDK Java Development Kit

events etc. The explicit introspection function uses the *BeanInfo Interface*, which returns Descriptor-Objects containing the requested information.

Customisation: Existing beans can be visually modified and adopted to address changed or new requirements. Extension are possible using the object oriented approach of polymorphism and encapsulation.

Event-Handling: Beans may exchange "Events" using the Delegation-Event-Handling-Model (e. g. Event-Sources and Event-Listener).

Methods: are used to change properties of the bean. In general, all methods declared as "public" are (besides the properties) accessible from outside the bean. Using a BeanInfo-class allows to define which methods will actually be available for the user later. Bean-Builder-Tools allow to customise the access restriction of Methods.

Properties: The "Properties" of beans can be changed easily. A certain syntax is required to make sure that GUI builder tools recognise the properties. This Syntax is called “Signature Patterns”. There are normal, indexed, bound and constrained properties. Normal properties represent a single value, indexed properties an array of values. Bound properties link a property with an event which will be triggered for instance if the value of the property changes. Constraint properties allow to require other beans to confirm changes in the property value.

Persistency: is required to permanently store a bean with all its properties to allow the use and configuration by visual bean builder tools.

3.5 Comparison of the approaches

The following empirical evaluation uses the criteria's for software component defined by our own an furthermore suggested in the paper to [8].

Issue	**TMN-MO**	**IN-SIB (CS1R)**	**Java-Beans**
Independent Functionality (Plug & Play)	good	very good	very good
Clarity of interface definition	very good	very good	good
Communication ability (Interaction)	poor[3]	poor[4]	good
Low Coupling	good	good	very good
Platform independent Problem view IT view	 very good poor	 very good poor[5]	 good[6] very good
Quality factors (ISO 9126) Portability Usability Efficiency Functionality	 good good n/a n/a	 poor good n/a n/a	 good good n/a good

[3] highly domain specific
[4] highly domain specific
[5] company specific adaptions
[6] no general standard available

Issue	TMN-MO	IN-SIB (CS1R)	Java-Beans
Reliability	n/a	n/a	n/a
Maintainability	poor	poor	poor
Description of components			
Graphical representation	yes	yes[7]	yes[8]
Textual description	yes	yes	yes
Formal language used	yes	yes (IDL)	no
Intuitive to handle	good	good	good
Standardisation	very good	very good[9]	good
Domain specific	very good	very good	good
Ease of documentation	poor	good	poor
Coupling pattern	n/a	available[10]	available
Hierarchy	n/a	not available[11]	available[12]
Granularity	optimum	not optimum[13]	optimum

Table 2: Comparison of component design approaches

4 Evaluation of Assets using Metrics

4.1 Measurement Goals

The ultimate goal is assessing reusability using measurement. One the one hand it would be desirable to identify properties which have a positive impact on re-use by the means of metrics. This information can be used to define acceptance criteria for the implementation of components which can be controlled by metrics. Such a set of acceptance criteria can also be used to identify component candidates in existing legacy systems. Table 3 shows a hypothesis established by Jones in [9] on the link between software errors and re-use. The question with this data is: how to identify an error-free component, thus again stressing the need for a metrics based quality assessment.

	Defects with 0% Reuse	Defects with 25% Reuse	Defects with 50% Reuse	Defects with 75% Reuse
Requirement	1.00	0.75	0.50	0.25
Design	1.25	0.94	0.63	0.31
Source Code	1.75	1.31	0.88	0.44
...				

Table 3: Correlation between errors and reuse [8]

4.2 Preconditions to use Software Metrics

[7] firm standard
[8] firm standard (SUN)
[9] more than one standardisation organisation (ITU, ETSI)
[10] rules and constraints exist, but no pattern
[11] available since version CS2
[12] aus Beans können wiederum Beans erstellt werden (kein implementierungsunabhängiges Modell)
[13] finer granularity disirable

For the efficient application of Software Metrics [3] suggests a framework for the establishment of measurement programs. This framework uses the idea of [5] to group the measurement artefacts into product, process and resource.

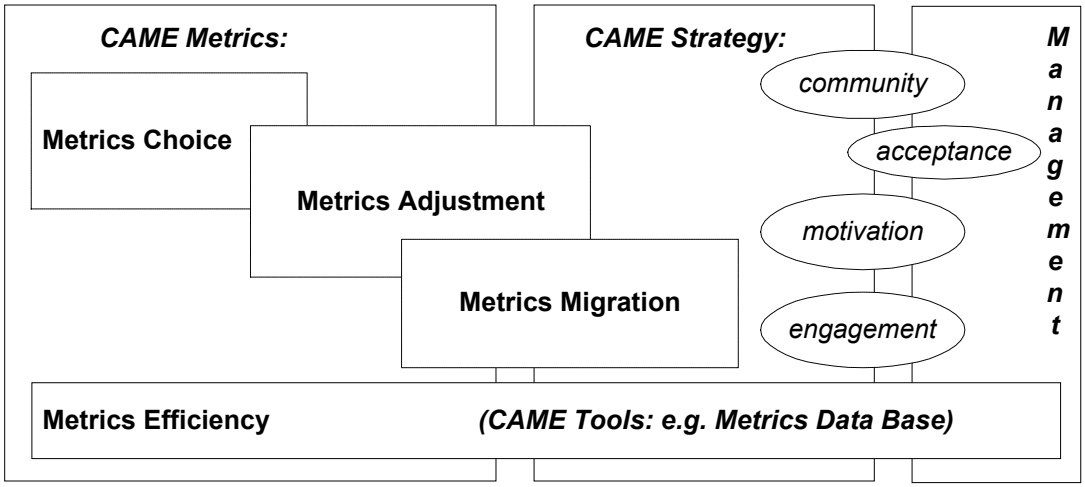

Figure 2: CAME Framework

The Framework itself is constituted by a CAME-Strategy (C-Community; A-Acceptance; M-Motivation; E-Engagement), a CAME-Framework (C-Choice; A-Adjustment; M-Migration; E-Efficiency) and CAME-Tools (C-Computer; A-Assisted; M-Measurement; E-Evaluation). The CAME-strategy focuses on aspects like the availability of a skilled group of metrics experts and their roles and responsibilities in the implementation of a metrics program. The CAME-Framework supports the selection of metrics, the identification of the properties of the used metrics, the efficiency of the metrics application and an estimation of the completeness of the selected set of metrics. Since the CAME-Strategy is already in place at the Deutsche Telekom the main objective is to use the CAME-Framework to select candidate metrics and assess their applicability in the re-use process. The following criteria will be used:

- **Reasonability** - is the metrics selected really measuring what is intended? A useful method to answer this question is the **G**oal-**Q**uestion-**M**etric Method ([1])
- **Completeness** - to which extend do the chosen metrics cover the properties of the artefacts produced during the different phases of the product life cycle
- **Effort Minimization** - are tools available to collect, store and retrieve the measurement values. An extensive automation is here required.
- **Empirical Evaluation** - is (internal or external) knowledge available to interpret the measurement results.
- **Data security & safety** - preferably data should be anonymous whenever possible. Only priviledged staff should have access to the data.
- **Scale type** (nominal, ordinal, intervall, ratio) - should be known for all used metrics to identify permissable statistical operations.
- **Persistency** – metrics values must be stored for later analysis.

4.3 Empirical Evaluation of available Bean libraries

The evaluation of the component technologies TMN-MO, IN-SIB and JavaBeans has been done using ordinal measures, i. e. providing a ranking. The "measurement values" have been

assigned by a team of application experts and are thus biased by the experience of the respective team member. In general we find that the components exhibit a good level of reusability with respect to some of the defined issues. There is however the need to further investigate into quality issues according to ISO 9126. The evaluation model has to be detailed and the domain specific knowledge has to be extended to other domains to approve the general applicability of the evaluation model.

As a first step we are currently performing an empirical evaluation using the source code of available JavaBean libraries. Our aim is to identify and define methods and solutions to develop components which are "certified" in accordance with the ISO 9126 quality criteria. Before re-using a component a developer should be able to assess the qualitative properties of the components by using software metrics. The following gives some requirements for meaningful metric sets:

- Metrics which observe the successful reuse for other projects (e.g. a component was already used 10 times).
- Static source code metrics for JavaBean properties:
  - Component size in Java Bytecode,
  - Component size in Java-Source code (e.g. LinesOfCode),
  - Complexity (e.g. components should have a low module complexity),
  - Comment density,
  - Coupling metrics
    - Low rates of fan/in, fan-out,
    - Clear structured input and output parameters,
  - Package and class count for a component. A Bean ought at least 3 classes contained
    - **THE BEAN ITSELF,**
    - **A BEANINFO-CLASS,**
    - **AND A CUSTOMIZER AND/OR EDITOR CLASS.**
  - Level of abstractness.
- Test coverage metrics,
- Error statistics,
- Performance metrics (e.g. response time, throughput, resource consumption).

Since JavaBeans are executable using portable bytecode it is not necessary to use metrics to assess the portability of a bean.

Measured value x		PACKAGE	METRICS OF THE BEAN-CLASS				
		Efferent Coupling	**Inheritance Deep**	**% Part of public Methods**	**Maximal McCabe of a Method**	**% Part comments**	**eLOC**
BDK[14] 12 Beans	x_{min}	2	0	57	1	2,7	17
	x_{max}	8	2	100	15	62,4	403
	$\bar{x}_{BDK}$	2,91	0,92	77	4,66	29,9	101,9
Standard Library	x_{min}	-	0	80	-	-	-
	x_{max}	-	0	90	-	-	-

[14] BDK – Bean Development Kit

5 Beans	$\bar{x}_{\text{Standard}}$	3	0	84	-	-	-
Prototype EJB[15] develop-ment 6 Beans	x_{min}	3	0	90	2	4,3	59
	x_{max}	7	0	100	11	24,2	255
	$\bar{x}_{\text{EJB}}$	5	0	97	4,66	13,6	118,49
23 Beans	$\frac{1}{23}\sum_{i=1}^{23} x_i$	**3,5**	**0,48**	**86**	**4,66**	**24,4**	**112,8**

Table 4: Selected results of an initial measurement of Beans

The 23 measured Beans from three different application domains, shows among other things, the following trends (selected examples):

- No abstract classes and only a very small inheritance depth are used. This is an indication that currently developed components consider technical aspects more than business aspects. Furthermore it shows weaknesses in the design of the components.
- The average value of the McCabe metrics (cyclomatic complexity, number of independent paths - here a method) is controlled by approx. 1,5. The maximum values of the McCabe metrics are represented in table 4, in each case referring to the corresponding methods of a class.
- Efferent Coupling as the number of classes outside the Java-package which are depended upon by classes within the package. High values are an identification, that the Bean has high relations to other components. That means, the JavaBean is not an independent component.
- Unexpected was the high part of methods defined as public. This visibility is especially a problem for JavaBean-components, because the Bean-Builder tools will import all public methods. By using the BeanInfo-class, it is possible to reduce the available methods for the Bean-Builder tools. The high number of public methods shows, that the tasks of maintenance were not considered within the development-phases. This statement is valid for libraries available at the market and for own developments too.
- The employed attributes of the 23 Beans were defined as private almost without exception. This corresponds to the principle of encapsulation in the object oriented paradigm and helps to prevent "side effects".

The expenditures used in connection with the prototype EJB evolution are also interesting. These efforts are not comparable with the previous experiences measured by function points or object points. The expenditures refer less to the steps of the development of components. Especially the deployment-step, the configuration and administration of the used application servers are time-intensive.

5 Conclusion and Outlook

The results presented here are just a first step in an ongoing research project. Especially the approach in comparing the well established concepts of TMN-MO and IN-SIBs with JavaBeans needs further research. We hope in return to use the experience of over 10+ years

[15] EJB – Enterprise Java Beans

on the former methods and apply this knowledge to JavaBeans. Besides metrics describing mainly the internal characteristics of components we also expect to be able to assess the software architecture and the application domain as well, since we expect both of them to have a high impact on reusability.

Emphases of further investigations refer to the identification of the "correct size" of a component, the definition of necessary processes during the composition (inclusive quality protection) and the guarantee from "Quality of Service" of the component (inclusive performance behaviour) themselves.

Based on this assessment we will establish programming guidelines (initially for JavaBeans) which will include threshold values for acceptable component properties.

References

[1] *Basili, V., Rombach, D.:* The TAME-Project: Towards Improvement-Oriented Software Engineering. IEEE Trans. on Software Engineering, 14(1988)6, S. 758 – 773

[2] *Buschmann, F., Meunier, R., Rohnert, H., Sommerlad, P., Stal, M.:* Pattern-orientierte Software-Architektur. Addison-Wesley, 1998

[3] *Dumke, R., Foltin, E., Winkler, A.S.:* A Framework for Object-Oriented Software Measurement and Evaluation. Proceedings of the IASTED Conference on Software Engineering, Oct. 28-31, 1998, Las Vegas, S. 129 - 132

[4] *Ezran, M., Morisio, M., Tully, C.:* Practical Software Reuse: The essential Guide. Paris: Freelife Publ., 1998

[5] *Fenton, N.:* Software Metrics - A Rigourous Approach. London: Chapmann & Hall Inc., 1991

[6] *Foltin, E., Schmietendorf, A., Dumke, R.:* Requirements and Design of an industrial Metrics Database. CONQUEST'99, Nürnberg, 27- 29. September 1999, S. 145-153

[7] *Gamma, E., Helm, R., Johnson, R., Vissides, J.:* Entwurfsmuster. Elemente wiederverwendbarer objektorientierter Software, Addison-Wesley, 1996

[8] *Griffel F.:* Componentware – Konzepte und Techniken eines Softwareparadigmas, dpunkt-Verlag, Heidelberg 1998

[9] *Jones, C.:* Software Quality Analysis and Guidelines for Success, Thomson Computer Press, 1998

[10] *Coplien, J.O., Schmidt, D.C. (ed.):* Pattern Languages of Program Design, Addison-Wesleay, 1994

[11] *Poulin, J.S.:* Measuring Software Reuse. Principles, Practices, and Economic Models. Reading/MA: Addison-Wesley 1997

[12] *Schmietendorf, A., Dumke, R., Foltin, E., Dimitrov, E., Wipprecht, M.:* Conception and Experience of Metrics-Based Software Reuse in Practice, Proceedings of IWSM99, 9th

International Workshop on Software Metrics, Sept. 08-09, 1999, Montreal, Canada, pp. 178-189

[13] *Schmietendorf, A., Dumke, R., Stojanov, S.:* Komponenten-orientierte Entwicklung verteilter Multiagenten-Applikationen, in Tagungsband 2. Workshop komponentenorientierte betriebliche Anwendungssysteme, Feb. 24-25, 2000, Wien, Österreich, S. 69-79

[14] *Sodhi, J., Sodhi, P.:* Software Reuse. Domain Analysis and Design Processes. New York ...: Mc Graw-Hill 1998

[15] *Udell, J.:* Component software, BYTE, 19(5):46-55, 1994

CALL FOR BOOK CHAPTERS

PERFORMANCE ENGINEERING WITHIN THE SOFTWARE DEVELOPMENT. PRACTICES, TECHNIQUES, AND TECHNOLOGIES.

A book edited by
Reiner Dumke, Claus Rautenstrauch, Andreas Schmietendorf, André Scholz
University of Magdeburg, Germany
Published by Deutscher Universitätsverlag

The objective of this book is to describe opportunities for performance engineering using innovative development practices, tools, and techniques. Each chapter will provide insight into the effective use of performance engineering through models, case studies, experience reports, or experiments. An emphasis is placed on organizational and management issues associated with the use of such technology including lessons learned and best practices. The book will be divided into three parts:

I. Relations between Software and Performance Engineering
II. Performance Models and Performance Measurement
III. Performance Engineering for Business Applications

Coverage (for illustrative purposes -- not limited to the following):

PART I:
- Performance Engineering Processes
- Integration Methods
- Performance Metrics

PART II:
- Performance Models
- Measurement Processes
- Tool Support

PART III:
- Case Studies
- Cost-/Benefit-Analysis
- Soft Human Factors (User Teaching, Culture Change..)

Submission Process

Authors are invited to submit a 2 - 5 page abstract that identifies the chapter's objective and explains its contents. Chapter abstracts are to be submitted by **September 1, 2000**. Authors will be notified by October 13, 2000, regarding the status of their chapter proposals. Full chapters of accepted abstracts are to be submitted by January 2, 2001.

Chapter abstracts are to be submitted electronically to:
ascholz@iti.cs.uni-magdeburg.de

Inquiries should be sent to:

André Scholz
Dep. of Computer Science
Business Information Systems
University of Magdeburg
Universitätsplatz 2
Magdeburg, 39106
Germany

The Evaluation of Customer Satisfaction by COSAM

Cornelius Wille

University of Magdeburg, Faculty of Computer Science,
Postfach 4120, D-39016 Magdeburg, Germany
Tel.: +49-391-6712812, Fax: +49-391-6712810, Email: wille@ivs.cs.uni-magdeburg.de

1 Introduction

In the measurement approach, we will chose and classify the customer intentions as an empirical basis for satisfaction evaluation. On the other hand, we will try to map software metrics to the empirical criteria to perform some investigations about the software process characteristics and the customer satisfaction and, finally, to work out a metrics-based measurement of the customer satisfaction in the IT area.

This three intentions are described in a simplified manner in Figure 1. The investigations of the mapping between satisfaction aspects and metrics are denoted by *adjustment* of the metrics values related to different ordinal values of the traditional customer satisfaction evaluation.

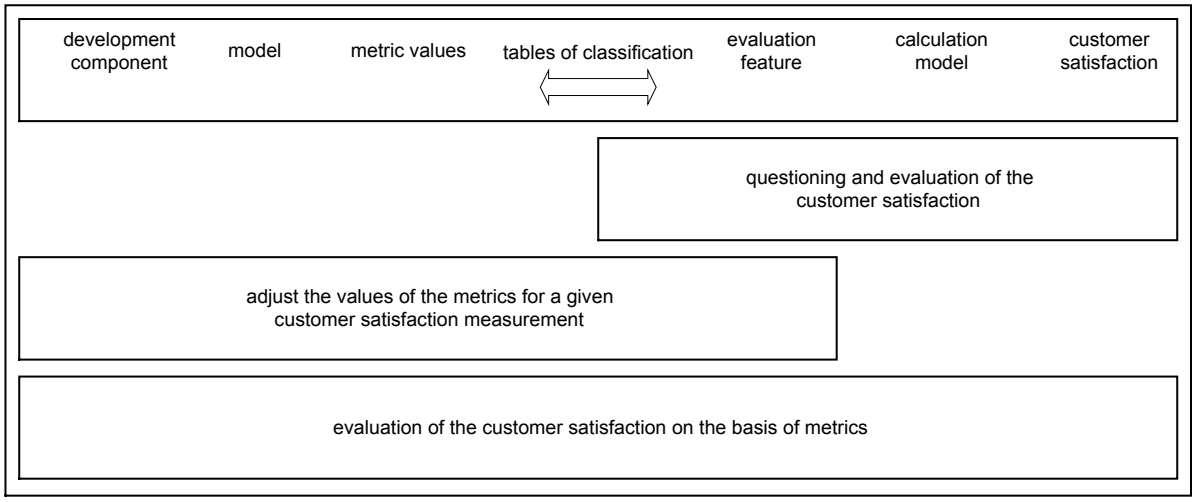

Figure 1: The three kinds of application of our approach for customer satisfaction evaluation

THEREFORE, WE HAVE DEFINED A SET OF ASPECTS FOR THE TRADITIONAL EMPIRICAL EVALUATION. THESE CRITERIA AND THE OPTIONAL KINDS OF SOFTWARE METRICS FOR MAPPING ARE SHOWN IN FIGURE 2 (BASED ON [DUMKE 99]). IN ORDER TO KEEP THE TRADITIONAL FORM OF THE CUSTOMER SATISFACTION EVALUATION, WE IMPLEMENTED THE FOLLOWING THREE METHODS (SEE ALSO [HAYES 92] AND [TALLEY 91]):

- The ***customer satisfaction index*** by [Mellis 98] as

$$KZI = \sum_{i=1}^{n} (W_i * Z_i)$$

with W_i as feature weight for feature i; and Z_i as satisfaction value for the feature i. The KZI considers the weights of the evaluation features apart from the satisfaction also. It can serve as yardstick for the comparison of the current with past test results of the same product/project or for the comparison of the examined product/project with others.

- The ***weighted total satisfaction*** by [Scharnbacher 98] considers likewise the satisfaction and the weights of the evaluation features.

$$Z_{ges} = \frac{\sum_{k=1}^{22} \frac{\sum_{i=1}^{n} \left(EZ_{ik} - \frac{|EZ_{ik} - W_{ik}|}{7} \right)}{n}}{22}$$

with EZ_{ik} as satisfaction of the customer i with feature k and W_{ik} as weighting of the feature k by the customer i.

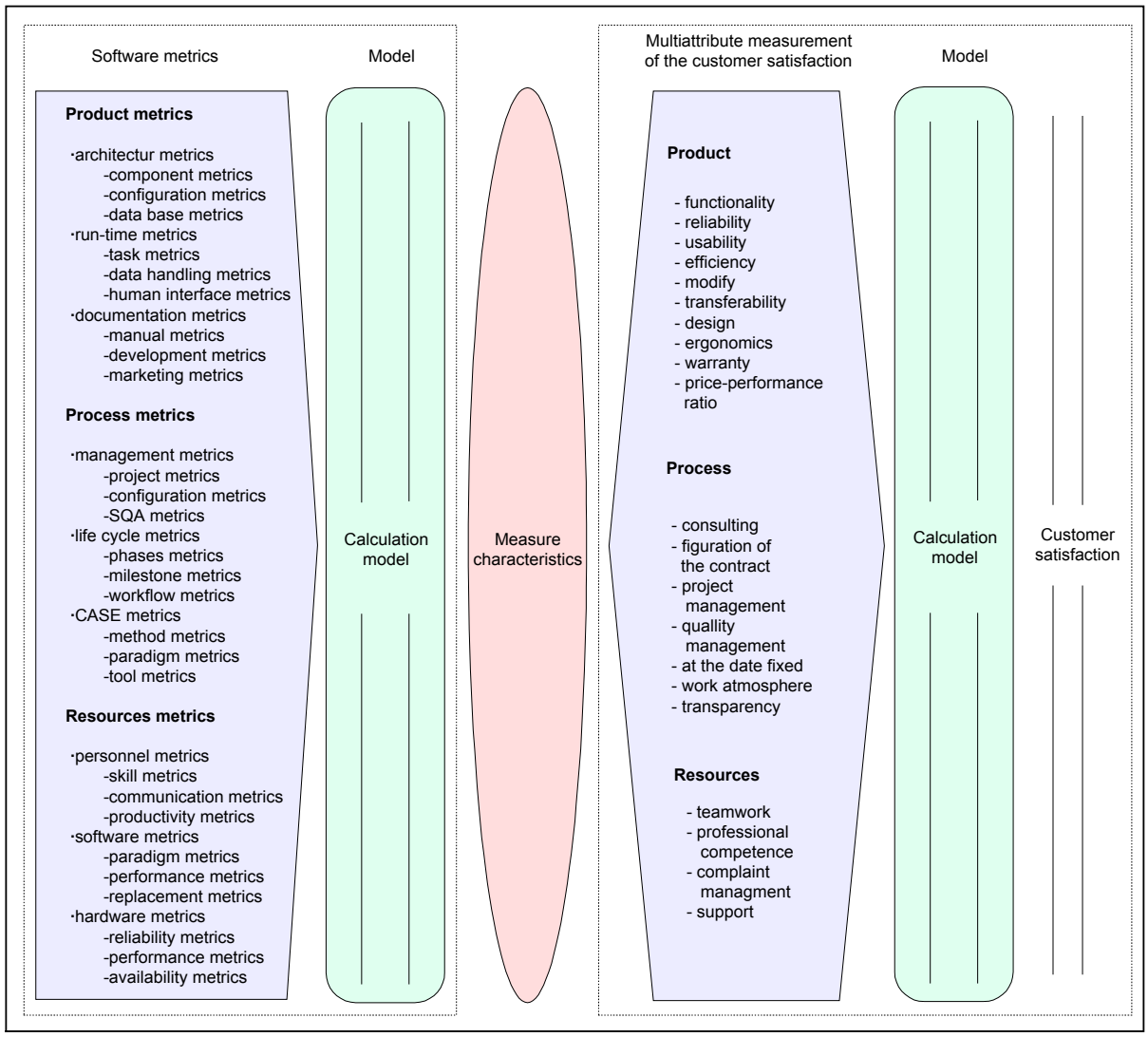

Figure 2: The empirical criteria and the possible metrics for mapping

- The ***customer satisfaction index*** by [Simon 98] as

$$KUZ = 1/n \sum_{i=1}^{n} Z_i$$

with Z_i as satisfaction value for feature i.

For the measurement of the customer satisfaction, we have chosen or defined as a first approximation *one metric* for *one empirical criterion.* Table 1 includes a general description of this kind of mapping. On the other hand, it is necessary to map the possible metrics values to the ordinal scale of the empirical criterion. We have chosen a unified ordinal scale for the empirical criterion from *1 (worst)* to *5 (best).* This aspect is at most indeterminate in our approach. Hence, its tool support requires a high flexibility for the adjustment or tuning of the measurement process to reflect the customer satisfaction determination.

Empirical criterion	Software metric
Functionality	*Traceability measure as relation between specified requirements*

	and the given requirements in the problem definition
Reliability	*Mean Time To Failure (MMTF)*
Usability	*Completeness of the documented product components*
Efficiency	*Performance of response time of the considered systems*
Modify	*Neighbourliness of maintenance*
Transferability	*Portability as relation between the effort of new development and the effort for transformation and adaptation*
Design	*Topology equivalence of the designed screens*
Ergonomics	*The level of the help support as number of online-documented system components*
Warranty	*Guarantee of software components in years*
Price-performance ratio	*Price level of the software related to the current market level*
Consulting	*Mean Time To Repair (MTTR)*
Figuration of the contract	*ISO 9000 Assessment/Certification*
Project management	*Capability Maturity Model evaluation*
Quality management	*ISO 9000 Assessment/Certification*
Complaint management	*Frequency of complaints per week*
At the date fixed	*Average deviation from milestone dates*
Support	*Availability of service in percent of the used cases*
Work atmosphere	*Environmental factor as relation between working hours without any break to the total working hours on the day*
Transparency	*Transparency of process as relation between manageable process components to the total number of process components*
Teamwork	*Provision of time as relation between the time spent to the necessary time for communication of every team member*
Professional competence	*Years of experience as level of developer skills*

Table 1: Mapping of software metrics to the empirical criteria for customer satisfaction evaluation

Based on this first step of mapping software metrics to the empirical aspects of customer satisfaction, we have defined a default mapping table in order to have a possibility to determine the customer satisfaction based on the different intervals of the metrics values.

2 The COSAM Tool

Our intention of the tool support is based on two general aspects:

1. The implementation of a client/server system to improve the current manual technique for the traditional customer satisfaction evaluation.

2. The implementation of a new approach of the measurement of customer satisfaction based of widely used software metrics.

The general user profile of the implemented system for customer satisfaction evaluation – the COSAM tool (see [Wille 00]) – is shown in Figure 3.

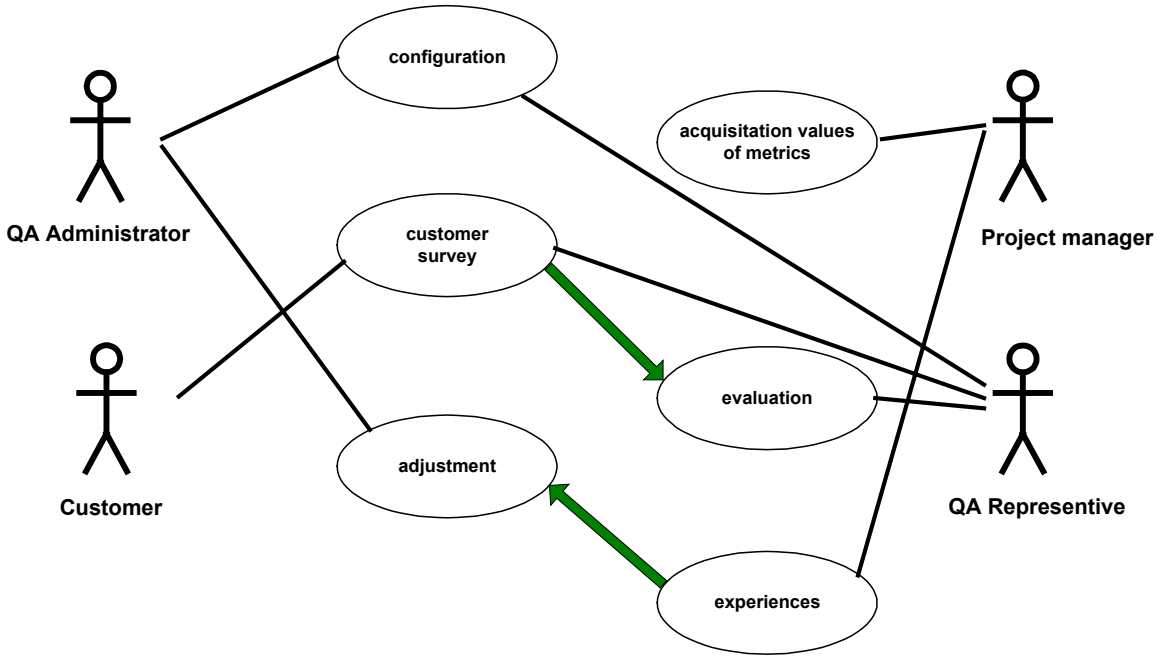

Figure 3: The user profile of the COSAM tool

On the other hand, the COSAM tool was implemented as a distributed system in order to perform a Web-based system application. The authenticity is kept by a special login technique. Figure 4 gives an general overview about the COSAM architecture.

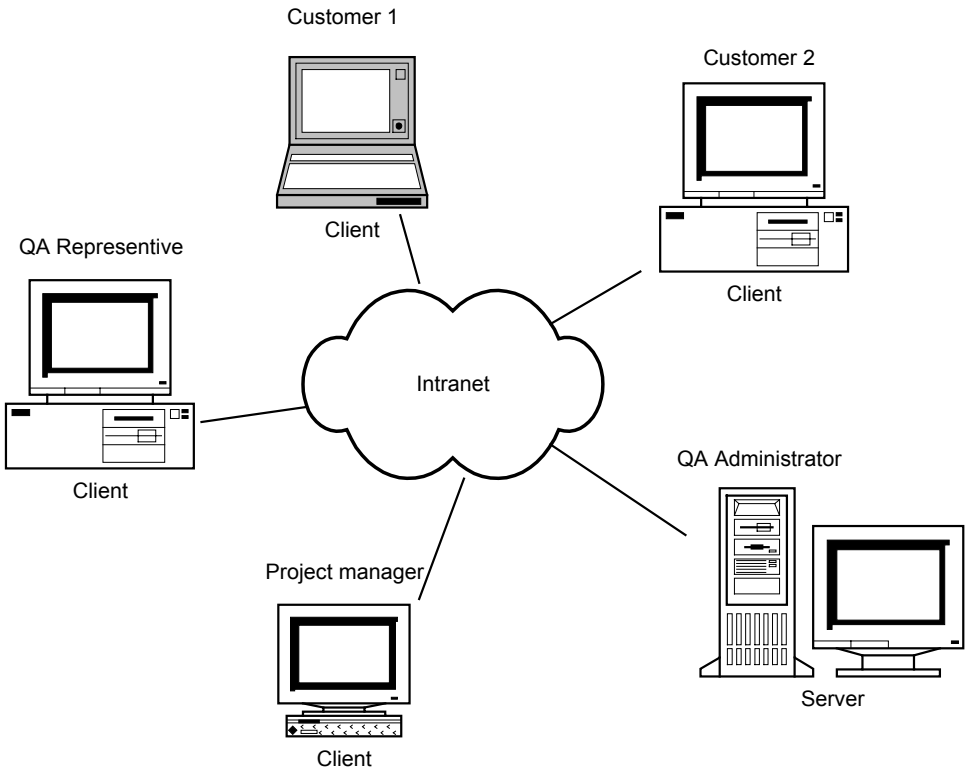

Figure 4: The distributed architecture of the COSAM tool

The COSAM tool allows to use different kinds of traditional customer satisfaction evaluation by choosing the empirical aspects and the evaluation method.

On the other hand, we can perform a metrics-based evaluation. The metrics values must be recorded manually in a screen, which is shown in Figure 5.

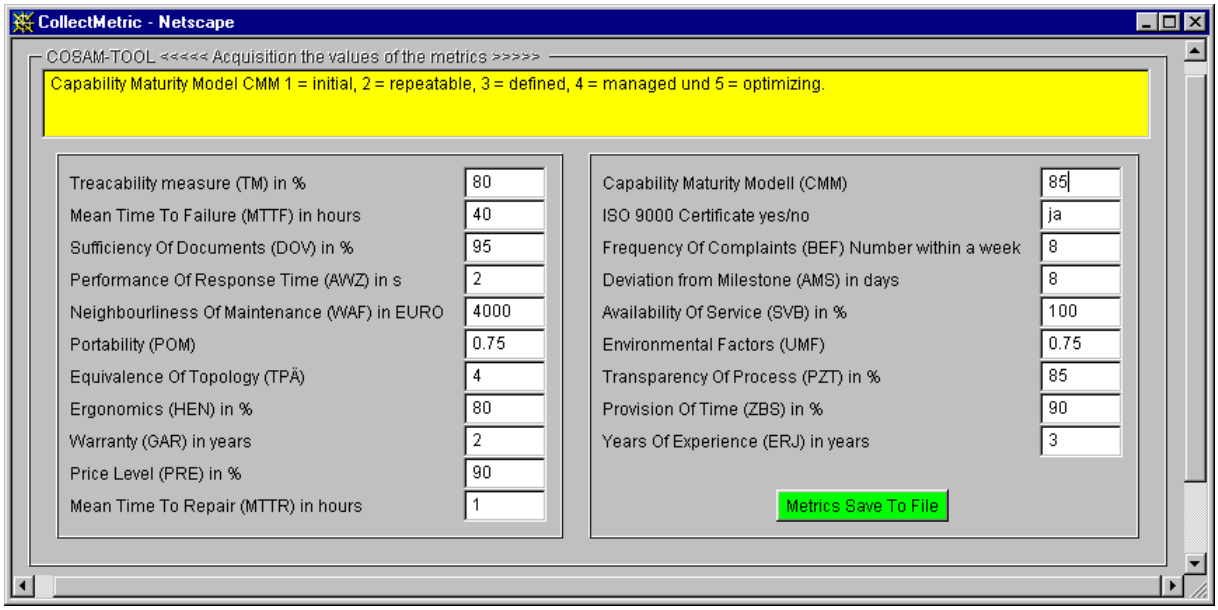

Figure 5: The COSAM screen for metrics value recording

Based on this recording, we can carry out the customer satisfaction evaluation by using one of the three implemented evaluation methods. Other screens are usable for changing or adaptation the mapping table or the choice of the empirical criteria in order to evaluate the customer satisfaction.

The COSAM is available for download at the URL

http://ivs.cs.uni-magdeburg.de/~wille/

References

[Dumke 99] Dumke, R.; Foltin, E.: *An Object-Oriented Software Measurement and Evaluation Framework.* Proc. of the FESMA, October 4-8, 1999, Amsterdam, pp. 59-68

[Hayes 92], Hayes, B. E.: *Measuring Customer Satisfaction.* ASQC Quality Press, Milwaukee, USA, 1992

[Mellis 98] Mellis, W.; Herzwurm, G.; Stelzer, D.: *TQM der Softwareentwicklung.* Vieweg Publ., Wiebaden, Germany, 1998

[Scharnbacher 98] Scharnbacher, K.; Kiefer, G.: *Kundenzufriedenheit: Ansalyse, Maßbarkeit und Zertifizierung.* Oldenbourg Publ. Munich, Germany, 1998

[Simon 98] Simon, H.; Homburg, C.: *Kundenzufriedenheit Konzepte Methoden Erfahrungen,* Gabler Publ., Wiesbaden, Germany, 1998

[Talley 91] Talley, D. J.: *Total Quality Management – Performance and Cost Measures.* ASQC Publ., Milwaukee, USA, 1991

[Wille 00] Wille, C.: *Konzeption und prototypische Implementation der Erfassung und Bewertung von Kundenzufriedenheit bei der Entwicklung, Wartung und Anwendung von Software.* Diploma Thesis, University of Magdeburg, Germany, March 2000

Process, Product and Resources Evaluation of Distributed Systems by the CORBA Metric Tool *CoMeT*

Makiola, H., Paschke, S.

DeTeCSM, Deutsche Telekom, Magdeburg, Germany/
SMLab, University of Magdeburg, Germany

1 Introduction

CURRENT MEASUREMENT TOOLS ARE ADDRESSED TO THE PRODUCT COMPONENTS OF CORBA BASED SYSTEMS ([BENATTOU 99], [MCGREGOR 98], [SCHMIETENDORF 99]). OUR SOFTWARE MEASUREMENT APPROACH FOR DISTRIBUTED SYSTEMS TRIES TO CONSIDER ALL THE SOFTWARE ENGINEERING ASPECTS AS PRODUCT, PROCESS AND RESOURCES [MAKIOLA 00]. THE MEASUREMENT INTENTIONS ARE BASED ON

- *THE PROCESS MEASUREMENT:* **THE *PROCESS EVALUATION COEFFICIENT (PBK)* IS THE MAIN METRIC USED FOR PROCESS EVALUATION. THIS METRIC IS CALCULATED AS**

$$PBK = (t_D \times A)/(n_D \times S)$$

with t_D - time of the product development, A – weighted application domain (e. g. health care product: 0.8, e-commerce system: 0.7, finance application: 0.65, telecom system: 0.6, transport application: 0.4), n_D – number of product developers, S – average skills of the developers (novice: 1, advanced: 3, expert: 5).

- ***the product measurement:*** The product evaluation is based on a first approximation of chosen object-oriented software metrics only. The metrics are described in Table 1.

Metric	**Measured contents**
AIF	Attribute inheritance factor of [Abreu 94]
ANA	Average number of attributes per class
ANK	Average number of classes per package
ANM	Average number of methods per class
ANP	Average number of parameters per method
COF	Coupling factor of [Abreu 94]
LOC	Lines of code (of method, class, package)
MIF	Method inheritance factor of [Abreu 94]

Table 1: Software metrics for the product evaluation

The thresholds for the evaluation are presented in a special kind of visualisation in the tool application indirectly.

- ***the resources measurement:*** The measurement of the software resources of the CORBA-based distributed systems was realised as measurement and evaluation of the applied CORBA by the development and system tool themselves.

2 The CoMeT-Based Measurement and Evaluation

The CoMeT tool has the following start page in order to begin the process and product measurement and evaluation (see Figure 1).

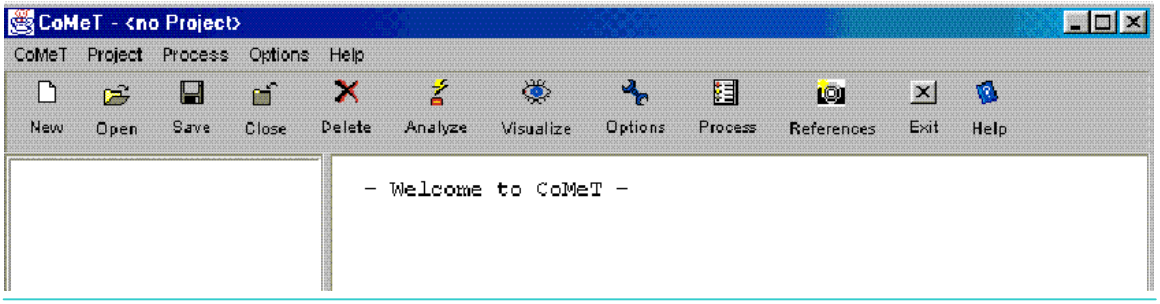

Figure 1: Start screen of the CoMeT tool

The process evaluation starts with the recording of all necessary information to execute the PBK factor. Figure 2 shows the consideration of the skill factor.

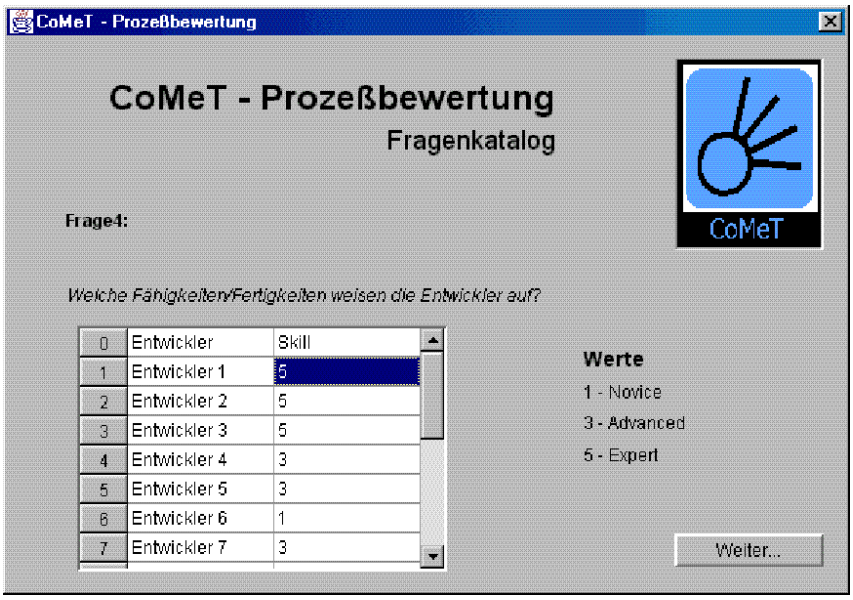

Figure 2: Recording the skill factor for the process evaluation

After the recording of all the four input characteristics, the CoMeT tool executes a process evaluation as shown in Figure 3.

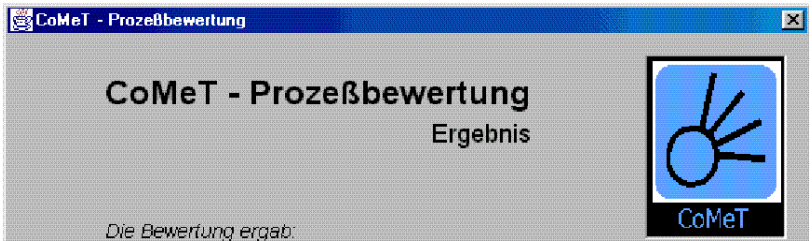

Figure 3: Process evaluation of CORBA-based systems

The product and resources evaluation is based on the chosen metrics described above. The CoMeT tool shows the current threshold intervals which can be changed (see Figure 4).

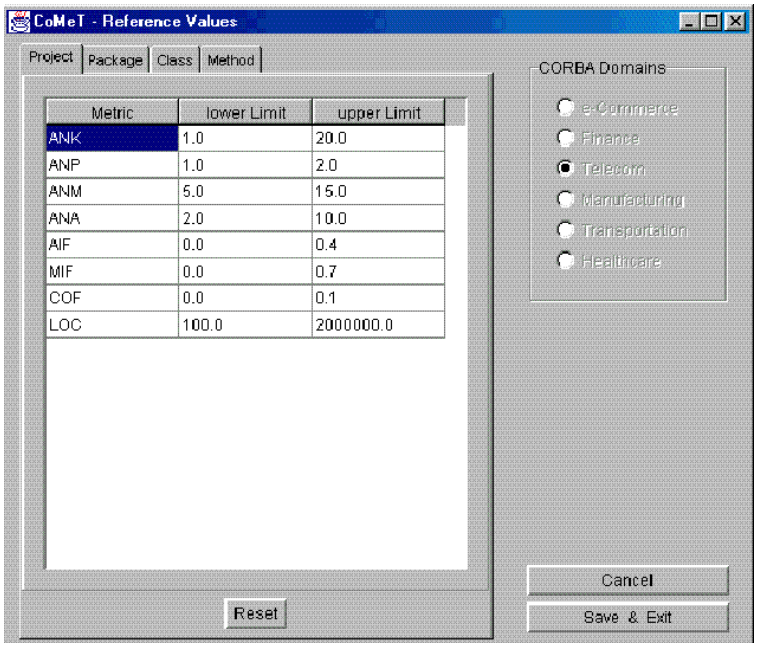

Figure 4: Current metrics thresholds for the product evaluation

On the other hand, the CoMeT tool allow a special kind of visualisation of the measured product characteristics based on the predefined software metrics. Figure 5 shows such an example of the product evaluation.

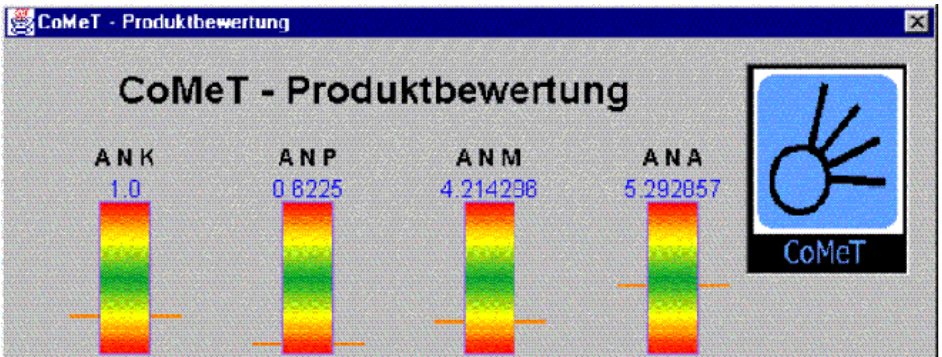

Figure 5: An example of product measurement visualisation by the CoMeT tool

The threshold areas are coloured in the product evaluation in both directions of lower and upper metrics values (green in the middle, yellow as warnings, and red at the problem areas). The product evaluation is related to the Java language as implementation paradigm for the CORBA-based distributed system and considers the different levels of architecture components as class, package and system.

The CoMeT tool is available for download at the SMLab URL at

http://ivs.cs.uni-magdeburg.de/sw-eng/us/

References

[Abreu 94] Abreu, F. B.; Carapuca, R.: *Candidate metrics for object-oriented software within a taxonomy framework*. Journal of Systems and Software, 26(1994), pp. 87-96

[Benattou 99] Benattou, M. et al.: *Principles and Tools for Testing Open Distributed Systems*. In: Csopaki et al.: Testing of Communicating Systems – Methods and Applications, Kluwer Academic Publ., 1999, pp. 77-92

[Makiola 00] Makiola, H.; Paschke, S.: *Evaluation of CORBA-based Software Systems with a new kind of Visualisation (German)*. Diploma Thesis, University of Magdeburg, June 2000

[McGregor 98] McGregor et al: *Collecting Metrics for CORBA-based Distributed Systems*. Proc. of the Fifth International Software Metrics Symposium, Bethesda, Maryland, Nov. 20-21, 1998, pp. 11-22

[Schmietendorf 99] Schmietendorf, A.: *Performance Engineering of Distributed Web Applications (German).* Diploma Thesis, University of Magdeburg, May 1999

Dumke, R.; Abran, A.: Software Measurement – Current Trends in Research and Practice

DUV Publisher, Wiesbaden, 1999 (269 pages)
ISBN 3-8244-6876-X

This new book includes key papers presented at the 8th International Workshop on Software Metrics in Magdeburg (Germany), September 1998. It is a collection of theoretical studies in the field of software measurement as well as experience reports on the application of software metrics in USA, Canadian, Netherlands, Belgian, France, England and German companies and universities. Some of these papers and reports describe new software measurement applications and paradigms for knowledge-based techniques, test service evaluation, factor analysis discussions and neural-fuzzy applications. Other address the multimedia systems and discuss the application of the Function Point approach for real-time systems, the evaluation of Y2K metrics, or they include experience reports about the implementation of measurement programs in industrial environments.

Humphrey, W. S.: Introduction to the Personal Software Process℠

Addison-Wesley, 1999 (278 pages)
ISBN 0-201-54809-7

One of the major challenges software engineers face transcends designing and programming software applications; it is managing their own personal approach to the software engineering process - to overcome the "hacker" ethic and work more effectively, efficiently, and productively.
In this practical introduction to the basic disciplines of effective software engineering, Watts Humphrey, well-known author of the influential book, *Managing the Software Process*, brings his Personal Software Process℠ to a wide audience of students and professional programmers. This hands-on book provides practical exercises readers can use to improve their time-management and quality-assurance practices, skills that will help them do competent professional work and better apply their programming expertise for greater success in their careers. *Introduction to the Personal Software Process provides:*

- Help for software engineers at ALL levels - from students to experienced professional - to become far more effective, efficient, and productive by allowing them too manage their work habits and personal software management techniques.
- Advice and guidance from one of the world's leading software process and software quality experts.
- Practical exercises for improving personal skills.

Support materials for this book are available on the World Wide Web at **http://www.awl.com/cseng**. Materials include copies of the forms illustrated in the book, and spreadsheets for the exercises.

Abran, A.; Dumke, R.: Proceedings of the 9th International Workshop on Software Measurement (IWSM'99)

Lac Superiew, Quebec, Canada, September 8 - 10, 1999
available at: ***http://www.lrgl.uqam.ca/iwsm99/index2.html***

Dumke, R.; Lehner, F.: Software-Metriken - Entwicklungen, Werkzeuge und Anwendungsverfahren

DUV Publisher, Wiesbaden, 2000 (229 pages)
ISBN 3-8244-7120-5

The includes the papers of the 9th German Workshop on Software Measurement in Regensburg in September 1999. The contents is

Michael Jacobsen-Rey
Automated Software Inspection - ATTAINING NEW LEVELS OF SOFTWARE QUALITY

THOMAS FETCKE
TWO PROPERTIES OF FUNCTION POINT ANALYSIS

Erik Foltin, Reiner Dumke, Andreas Schmietendorf
ENTWURF EINER INDUSTRIELL NUTZBAREN METRIKEN-DATENBANK
Projekt: metricDB-2 V 0.8

Claus Lewerentz, Heinrich Rust, Frank Simon
QUALITY - METRICS - NUMBERS - CONSEQUENCES

Reiner Dumke
Erfahrungen in der Anwendung eines allgemeinen objekt-orientierten Measurement Framework

Andreas Schmietendorf, Evgeni Dimitrov, Reiner Dumke, Erik Foltin, Michael Wipprecht
Konzeption und erste Erfahrungen einer metrikenbasierten Software-Wiederverwendung

Patricia Mandl-Striegnitz
UNTERSUCHUNG EINES NEUEN ANSATZES ZUR PROJEKTMANAGEMENT-AUSBILDUNG

Hans Windpassinger
MÖGLICHKEITEN DER METRIK-BASIERTEN MODELLIERUNG UND AUSWERTUNG VON QUALITÄTSVORGABEN MIT DEM WERKZEUG LOGISCOPE

Silvio Löffler, Frank Simon
SEMIAUTOMATISCHE, KOHÄSIONSBASIERTE SUBSYSTEMBILDUNG

Ulrich Schweikl, Stefan Weber, Erik Foltin, Reiner Dumke
Applicability of Full Function Points at Siemens AT

Harry M. Sneed
Testmetriken für objektorientierte Bankenanwendungen

Christof Ebert
Process Change Management in Large Organizations

Angelika Mittelmann
Messen von weichen Faktoren - EIN ERFAHRUNGSBERICHT

Humphrey, W. S.: Introduction to the Team Software Process[SM]

Addison-Wesley Longman, Inc., 2000 (463 pages)
ISBN 0-201-47719-X

Watts Humphrey is the visionary behind the Capability Maturity Model (CMM)® and The Personal Software Process (PSP)[SM]. The CMM contains a framework for software process improvement at the organizational level. The PSP builds the self-discipline needed for individual programmers to work efficiently and effectively. The author's new Team Software

Process (TSP)[SM] details methods to guide the formation of software development teams, to motivate their work, and to enhance their productivity.
This book describes an introductory version of TSP, ideal for smaller projects but also useful for learning basic techniques and procedures that apply to other development projects. Methods presented include:

- how to establish roles;
- how to conceive, design, and plan a project; and
- how to track and report on progress.

The book walks readers through a complete development cycle, illustrating:

- how best to use the talents at hand;
- how to formulate well-defined goals;
- how to coordinate activities for maximum progress;
- how to promote effective communication; and
- how to alleviate many of the conflicts that undermine teamwork.

Team members should not have to expend valuable time and energy reinventing ways to organize and run their team. By following a proven process, the team will more quickly be able to focus on the successful completion of the project itself. To help a team course apply these methods, the book provides two project exercises with prescribed development goals and team roles.

Wohlin,Claes et al: Experimentation in Software Engineering - An Introduction

Kluwer Academic Publishers Boston/Dordrecht/London, 2000 (204 pages)
ISBN 0-7923-8682-5

The purpose of EXPERIMENTATION IN SOFTWARE ENGINEERING: *An Introduction* is to introduce students, teachers, researchers, and practitioners to experimentation and experimental evaluation with a focus on software engineering. The objective is, in particular, to provide guidelines for performing experiments evaluating methods, techniques and tools in software engineering. The introduction is provided through a process perspective. The focus is on the steps that must be taken to perform experiments and quasi-experiments. The process also includes other types of empirical studies.
The motivation for the book emerged from the need for support the authors experienced when making their software engineering research more experimental. Several books are available that either treat the subject in very general terms or focus on some specific part of experimentation; most focus on the statistical methods in experimentation. These are important, but there are few books elaborating on experimentation from a process perspective; none addressing experimentation in software engineering in particular.
The scope of EXPERIMENTATION IN SOFTWARE ENGINEERING: *An Introduction* is primarily experiments in software engineering as a means for evaluating methods, techniques and tools. The book provides some information regarding empirical studies in general, including both case studies and surveys. The intention is to provide a brief understanding of these strategies and in particular to relate them to experimentation.
EXPERIMENTATION IN SOFTWARE ENGINEERING: *An Introduction* is suitable for use as a textbook or a secondary text for graduate courses, and for researchers and practitioners interested in an empirical approach to software engineering.

Belanger, F.; Jordan, D. H.: Evaluation and Implementation of Distance Learning: Technologies, Tools and Techniques

Idea Group Publishing Hershey (USA), London (UK), 2000 (246 pages)
ISBN 1-878289-63-2

The twentieth century has seen the creation and evolution of technologies beyond imagination a century ago. The computer has enabled the digital presentation of knowledge, and increased the speed with which information can be captured and processed. Communication technologies have made possible the storage, transfer and sharing of information across vast distances and different time zones.
The acceptance of these technologies has led to a new alternative for providing education and training - distance learning. This book focuses on the processes, techniques and tools that are being used to successfully plan, implement and operate distance learning projects. Some interesting metrics are defined in order to evaluate this kind of Web-based software systems. Both professionals and educators who must enter this challenging teaching and training environment in the new millennium will benefit from *Evaluation and Implementation of Distance Learning: Technologies, Tools and Techniques*.

Bundschuh, M.; Fabry, A.: Aufwandschätzung von IT-Projekten

MITP Publisher, Bonn, 2000 (available October 2000)

This new book about software effort and costs estimation, includes a description of the current used methods in practice. A detailed presentation considers the Function Point methods and their different approaches. The book includes some case studies and is directed for a general practical use in the IT area.

FMSP 2000:

International Symposium on Software Testing and Analysis
August 21 - 24, 2000, Portland, Oregon, USA
see: **http://www.ics.uci.edu/issta-fmsp**

IFPUG 2000, Fall:

International Function Point User Group Fall Conference
September 11 - 15 , 2000, San Diego, USA
see**: http://www.ifpug.org/conferences/conf.html**

CONQUEST 2000:

Conference on Quality Engineering in Software Technology
September 14 - 15, 2000, Nuremberg, Germany
see: **http://www.asqf.de/**

UML 2000:

Third International Conference on the Unified Modeling Language
October 2 - 4, 2000, York, UK
see: **http://www.cs.york.ac.uk/uml2000/**

FESMA 2000:

3rd Conference on European Federation of Software Metrics Associations
October 2 - 6, 2000, Madrid, Spain
see: **http://www.fesma.org/**

IWSM'2000:

10th International Workshop on Software Measurement
October 4 - 6, 2000, Berlin, Germany
see: **http://ivs.cs.uni-magdeburg.de/sw-eng/us/IWSM2000/**

ISSRE 2000:

Eleven International Symposium on Software Reliability Engineering
October 8 - 11, 2000, San Jose, California
see: **http://www.rstcorp.com/conferences/issre2000**

ICSM 2000:

International Symposium on Software Maintenance
October 11 - 14, 2000, San Jose, California
see: **http://www.rstcorp.com/conferences/icsm2000**

PNSQC 2000:

2000 Pacific Northwest Software Quality Conference
October 16 - 18, 2000, Portland, Oregon
see: **http://www.pnsqc.org**

APAQC 2000:

First Asia-Pacific Conference on Software Quality
October 30 - 31, 2000, Hong Kong, China
see: **http://www.csis.hku.hk/~apaqs/**

DASMA 2000:

Workshop of the German Federation of Software Measurement
November 30 - December 1, 2000, Düsseldorf

see: **http://www.dasma.de/**

EuroSTAR 2000:

8th European International Conference on Software Testing Analysis & Review
December 4 - 8, 2000, Copenhagen, Denmark
see: **http://www.eurostar.ie/**

CSMR'2001:

5th Euromicro Working Conference on Software Maintenance and Reengineering
March 14 - 16, 2001, Lisboa area, Portugal
see: **http://www.esw.inesc.pt/csmr2001**

METRICS 2001 & ESCOM 2001:

7th International Symposium on Software Metrics
April 2 - 6, 2001, London, England
see: **http://www.telecom.lth.se/**

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://fdd.gsfc.nasa.gov/seltext.html**
  The Software Engineering Laboratory at NASA/Goddard Space Flight Center. Some documents on software product and process improvements and findings from studies are available for download.

- **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://saturne.info.uqam.ca/Labo_Recherche/lrgl.html**
  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 at 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, having 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.aemes.fi.upm.es**
  AEMES Association Espanola de Metricas del Software

- **http://www.asqf.de**
  ASQF Arbeitskreis Software-Qualität Franken e.V., Nuremberg, Germany

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

- DANMET: Danish Software Metrics Association

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

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

- **http://www.fesma.org/**
  FESMA Federation of European Software Metrics Associations

- **http://www.sttf.fi**
  FiSMA Finnish Software Metrics Association
- FFPUG: French Function Point User Group

- FPUGA: Function Point User Group Austria

- **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.pitt.edu/~ddarcy/isprof/intotool.html#intro**
  Metrics Tool Listings by Dace Darcy

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

- **http://davidfrico.com/mettools.htm**
  Software Metrics Tools by Dave
- **http://mdmetric.com/meastl1.htm**
  Maryland Metrics Tools

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

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 5 *2000* *NUMBER 1*

CONTENTS

ISSN 1431-8008