dpunkt.verlag im Inter

Ina Schieferdecker ·
Stephan Goericke (eds.)

Setting Quality Standards

Since 1997 the international "Confer
Quality Engineering in Software Tec
(CONQUEST) has gathered together t
ware and quality engineering comm
discuss aspects of software quality
share experiences in industrial and
projects.

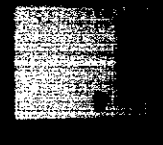
The book constitutes a selection of
sented speeches and lectures. They
first-hand information on the practi
application and ongoing developme
methods and techniques and provid
into real-life case studies along with
quality analysis and evaluation.

The proceedings of the CONQUEST
include

- Processes
- Experience Reports
- Next Generation Testbed for Medi
- Engineering
- Testing
- Secure Software Development
- QA for Finance IT
- Metrics
- Quality and Safety

CONQUEST 2008 is organized by th
national Software Quality Institute
in cooperation with Fraunhofer FO
the German Society for Informatic
kindly supported by the Ministry
mics of Brandenburg, the networ
rity and Safety made in Berlin-Br
(SeSamBB), and the ZukunftsAgen
denburg (ZAB).

ISBN 978-3-89864-567-6



Schieferdecker · Goericke
Setting Quality Standards

Ina Schieferdecker · Stephan Goericke (eds.)

Setting Quality Standards

Proceedings of the CONQUEST 2008

11th International Conference on
Quality Engineering in Software Technology

Potsdam 2008

dpunkt.verlag

Kindly supported by:

LAND BRANDENBURG
Ministerium für Wirtschaft

SeSamBB
Security and Safety made in Berlin-Brandenburg

ZAB ZukunftsAgentur Brandenburg

Ina Schieferdecker · Stephan Goericke (eds.)

Setting Quality Standards

Proceedings of the CONQUEST 2008

11th International Conference on
Quality Engineering in Software Technology
Potsdam 2008

The conference is organised by iSQI – International Software Quality Institute.

dpunkt.verlag

ASQF Arbeitskreis Software-Qualität und -Fortbildung e.V.

iSQI International Software Quality Institute

iSQI – International Software Quality Institute
Am Weichselgraben 19 · 91058 Erlangen · www.isqi.org

Ina Schieferdecker
ina.schieferdecker@fokus.fraunhofer.de

Stephan Goericke
stephan.goericke@isqi.org

Editor: Vanessa Wittmer
Copy-Editor: Julia Neumann, Prince George BC, Canada
Producer: Birgit Bäuerlein
Cover Design: Helmut Kraus, www.exclam.de
Printer: Koninkliijke Wöhrmann B.V., Zutphen, Netherland

Bibliografische Information Der Deutschen Bibliothek
Die Deutsche Bibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über <http://dnb.ddb.de> abrufbar.

ISBN 978-3-89864-567-6

1st Edition

Ringstraße 19 B
69115 Heidelberg
Germany

543210

Greetings

Stel Pinhasov Beck
Commercial Attaché of the Embassy of Israel and
Director of the Israel Trade Center:

Shalom,

Dear Participants,

Welcome to this year's CONQUEST in Potsdam with its special focus on Israel.

As the Economic Representative of the State of Israel in Germany, I am especially pleased that Israel has been chosen to be iSQI's official partner country at CONQUEST 2008.

This is a great opportunity to present Israel's outstanding technologies to the international software development community.

At the same time, this year's CONQUEST offers an excellent platform to further intensify the bilateral relations in the software and security sector between Germany and Israel.

Israel's software and security industries' reputation as one of the most innovative in the world and the enormous amount of requests to my office from respective companies looking for business partners in Germany already give an idea about the strong potential of future co-operation.

That is why the Israel Trade Center and the Israel Export and International Co-operation Institute have put great efforts into bringing a delegation of Israeli companies to this year's CONQUEST. In particular, the Brandenburg meets Israel BusinessLink with its pre-scheduled B2B meetings will enable experts to get to know each other and to exchange potential business ideas.

I wish all of us an inspiring and fruitful conference and I'm looking forward to meeting you in Potsdam!

Sincerely,
Stel Pinhasov Beck

Greetings

Prof. Ina Schieferdecker, Conference Chair:

I am pleased to welcome you to CONQUEST 2008 in September 24-26, 2008 in Potsdam, Germany. Since 1997, CONQUEST – the International Conference on Quality Engineering in Software Technology – has been the platform for software professionals to get to know the latest methods and tools for quality engineering from industry and research, to listen to practical experiences and show cases, to see how quality engineering methods are successfully deployed in an industrial environment, to learn about recent tool and service offers, to share experiences and to discuss future directions.

CONQUEST 2008 will feature tutorials and presentations of internationally renowned speakers and high-quality peer-reviewed presentations from members of the quality engineering community. It provides a full picture of software quality in theory and practice.

It is a particular pleasure to celebrate at CONQUEST the 10th anniversary of the Software Testing Working Group at ASQF, which was created back in 1998 and currently runs over 20 meetings a year with more than 500 participants. It also contributed to the Certified Tester Initiative in Germany which, together with more than 30 other national testing boards, can currently draw on more than 90.000 certificates worldwide. All working group members, certified testers and interested parties are cordially invited to the Software Tester Welcome, which is supported by the German Testing Board (GTB). In addition, the GTB-

Forum will take place for the first time in Potsdam as an independent, but affiliated event of CONQUEST.

A variety of international speakers and participants will give you practical and research insights into current and topical aspects of quality engineering. This broad perspective will be presented by attendees from all over the world including Europe, Asia and America. Our exhibition on recent methods, services and tools will include 15 national and international companies from the software quality field and will give you detailed insights into their tools and services.

I wish us all a successful and enjoyable CONQUEST 2008.

Yours truly,
Ina Schieferdecker

Greetings

Stephan Goericke,
Director iSQI:

As the Director of the International Software Quality Institute, I'm very pleased to present this documentation issue for the CONQUEST 2008. Meanwhile, it's now the 11th time that software engineers from all over the world have accepted iSQI's invitation to come to be informed and to exchange information on the topic of Software Quality Improvement. This year's new topic concerns itself with "security and safety" – on the one hand, the area of software development and on the other, secure and safe software-based systems.

The 11th CONQUEST offers another novelty, about which I'm particularly pleased. This year, Israel has become the partner country of the conference, to commemorate the occasion of the 60th anniversary of the founding of the State of Israel and the good business relations which iSQI has made in recent years in the high-tech Mediterranean state. Many representatives of Israeli businesses and institutions are involved in the CONQUEST 2008. It is an honour for us to be able to further enhance business relationships and the transfer of knowledge between the two countries. Only then can we reach our goal to better software quality and institute better standards worldwide.

iSQI now supplies certification in over 40 countries, standards now exist in the areas Software Test, TTCN-3, Software Architecture, Project Management, Innovation Management, Configuration Management, Requirements Engineering, the SPICE-Assessors Certification following iNTACS, IT Security Manage-

ment, Secure Software Engineering, Information Security and soon also the V-Model XT. In addition, the iSQI has developed a standard on the basis of these "certified" programs, which incorporates the recognition of many years of practical experience and which allows for a flexible adaption in an individual job situation: QAMP (Quality Assurance Management Professional). If you're interested in this standard, you'll learn more about it during the CONQUEST.

I'd like to thank all those partners who supported us during this CONQUEST in Potsdam and also all those who helped to make the CONQUEST what it is: one of the most important specialized events in the area of software quality.

I wish you all the best and great success in your work and I would be very pleased to meet you at our future conferences and educational programs.

Sincerely,
Stephan Goericke

Program Committee

Conference Chair:

Ina Schieferdecker, Fraunhofer FOKUS, Germany

Program Committee:

Girts Baltaisbrencis, Exigen Services, Latvia
Manfred Broy, Technical University Munich, Germany
Wim Demey, FRSGlobal, Belgium
Winfried Dulz, University of Nuremberg-Erlangen, Germany
Karol Frühauf, INFOGEM, Switzerland
Anne Mette Jonassen Hass, DELTA, Denmark
Georg Heidenreich, Siemens, Germany
Bernard Homès, TESSCO Group, France
Dehua Ju, ASTI Shanghai, China
Dieter Landes, University of Applied Sciences Coburg, Germany
Peter Liggesmeyer, Fraunhofer IESE, Germany
Alon Linetzki, Best-Testing, Israel
Patricia McQuaid, California Polytechnic State University, USA
Pedro Merino, University of Malaga, Spain
Ludger Meyer, Siemens, Germany
Joanna Nowakowska, Oracle, Poland
Mohammad Nuruzzaman, Daffodil International University, Bangladesh
Barbara Paech, University of Heidelberg, Germany
Sachar Paulus, SAP, Germany
Helmut Pichler, ANECON, Austria
Klaus Pohl, University of Duisburg-Essen, Germany
Ita Richardson, University of Limerick, Ireland
Bj Rollison, Microsoft, USA
Alexander Rudolph, Ecolab, Austria

Sergii Riapolov, Ukrainian Scientific Center for Development of Information Technologies (ITDEV) under Ministry of Education and Science of Ukraine, Ukraine
Achim Schmidt, Beta Systems, Germany
Kurt Schneider, University of Hannover, Germany
Andreas Spillner, University of Applied Sciences Bremen, Germany
Maria Stefanova-Pavlova, CITT Global, Bulgaria
Angela Vozella, CIRA, Italy
Mario Winter, University of Applied Sciences Cologne, Germany
Walter Wintersteiger, Management & Informatik, Austria
Peter Zimmerer, Siemens, Germany

Additional Reviewers:

Stefan Wagner, Technical University Munich, Germany
Eva Geisberger, Technical University Munich, Germany
Marco Kuhrmann, Technical University Munich, Germany
David Cruz, Technical University Munich, Germany
Heiko Stallbaum, University of Duisburg-Essen, Germany
Times Illes-Seifert, University of Heidelberg, Germany
Lars Borner, University of Heidelberg, Germany
Jürgen Rückert, University of Heidelberg, Germany
Shamsuddin Ahammad, Daffodil Institute of IT (DIIT), Bangladesh

Table of Contents

Software Product Quality: An Open Source Automated Measurement Environment

M. Rodríguez Monje · J. Garzás Parra

Kybele Consulting, Madrid, Spain

M. Piattini Velthuis

Departamento de Tecnologías y Sistemas de Informacíon (UCLM), Ciudad Real, Spain

Abstract

Establishing measurement systems is a basic piece of software quality control, ever more so, given the current trend of "outsourcing" software development, which is frequently carried out by external teams or software factories. It also has to be remembered, that to be efficient, a measurement system of software product quality must have a high automation level, which allows it to be used frequently, without consuming excessive time. In pursuit of such goals, we have designed KEMIS, a measurement environment for software product quality based on open source tools. KEMIS uses software tools, such as Maven 2 and its measurement and reporting plug-ins, which allow the automatic calculation of metrics, the customization of results and the definition of the periodicity of measurements. The chief goal of this paper is to describe the main characteristics of KEMIS.

1 Introduction

Nowadays, organizations are very interested in getting competitive advantages and are therefore investing in the automation of their business processes and in developing new services based on software technology. All this requires an increasing confidence in software products and a consideration of the strategic role that these have in the organizations. This has meant that an increase in the quantity of software products has become a vital necessity, as has quality control of these.

Software products are as sophisticated as they are complex; this is becoming increasingly so [Won 04] [Won 05], thus creating the challenge to develop these products within time restrictions, while not forgetting about quality.

To assure that the process or its products have the required quality, it is necessary to assign values, indicators or other mechanisms by means of which this evaluation can be carried out. So, we need a process of software measurement that pursues three main goals in general: to help us understand what happens during development and maintenance, to allow us to control project evolution and to improve the processes and products [Fen 97].

Using metrics is a good way to understand, check, control, predict and test software development and maintenance projects [Bri 96] and it can be used by professionals and investigators to take better decisions [Plf 97]. An environment that makes measurement possible requires both a methodological support and a technological support [Lav 00]. That being so, for these metrics to be used to evaluate the software products in a practical, efficient and exact way, we must use tools that automate the acquisition, presentation and analysis of the values obtained [Gil 95].

This paper puts forward the KEMIS project (Kybele Environment Measurement Information System)[1] which provides a free software environment for the acquisition, analysis and presentation of software product quality metrics in an automatic way. In addition, KEMIS presents a methodological support based on best measurement practices like Goal Question Metric (GQM), Goal Question Indicator Metric (GQ (I) M), Goal-Driven Software Measurement (GDSM), Practical Software Measurement (PSM), and it is perfectly aligned with specific measurement standards (ISO 15939, ISO 9126, IEEE STD 1061-1998, etc.) and with measurement activities of main software processes models (CMMI, ISO 12207, ISO 15504).

The remainder of the paper is organized as follows. Section 2 provides an overview of the KEMIS project, outlining its objectives and the technological and methodological support for those. Subsequently, section 3 describes the KEMIS architecture and phases of the environment implementation. Section 4 presents the key metrics and indicators of product quality studied in the KEMIS project. Section 5 shows the benefits of KEMIS for companies that outsource their development, as well as for software factories which make their own software products. Finally, section 6 presents a set of conclusions and recommendations for measuring software product quality, as well as the future research work of the KEMIS Project, which allow us to define and generate new indicators and reports.

1 http://www.kybeleconsulting.com/pages/kemisproject/kemisproject.html

2 Background of the KEMIS Project

KEMIS "Kybele Environment Measurement Information System" is an environment suggested by Kybele Consulting that provides, on the one hand, a predefined set of free software applications, along with their configuration and installation, which allows the implanting of a measurement system for software quality at operative, tactical and strategic levels. On the other hand, it gives methodological support for the evaluation of software product quality, based on PSM (Practical Software Measurement).

It is true that metrics, inspections, revisions and quality controls are classic activities that have years of maturity. But at the present time, to ensure that these become useful practices and to obtain broad benefits from them, we have to consider the following objectives:

1. To **define clear measurement objectives**, determining what data are necessary and avoiding risks, because an excess of information means that the final mission of the measurement program gets lost.
2. **To conduct the measurements regularly and frequently**, making it possible to take decisions at the opportune moment.
3. **To automate the measurement process**, making the previous objectives possible, since a measurement that takes a long time or which is very manual prevents the measurement process from being performed on time.
4. **To define different abstraction levels**, which prevents us from losing details and which shows, at all levels (operative, tactical and strategic), the precise information for making a decision.

We will go on to present the way in which KEMIS covers the above goals by means of methodologies, techniques and measurement tools that will facilitate the processes of companies in measuring the quality of their software products.

2.1 Methodological Support

In the quest to fulfill the first goal (to define clear measurement objectives), KEMIS is based on PSM "Practical Software and Measurement System" [McG 06]. PSM proposes a measurement process for managing the technical and business objectives of an organization, and incorporates best practices used by professionals when conducting measurement within software communities. As can be seen in Figure 1, PSM served as a basis for standard ISO/IEC 15939, Software Engineering – Software Measurement Process [ISO 02]. This standard describes the measurement process in terms of purpose and results of set activities and associated tasks. PSM provides additional details for the successful carrying out of the work presented in the ISO/IEC 15939. The purpose and results of the measurement process proposed in ISO/IEC 15939 have been added to ISO/IEC

12207 and ISO 90003: implementation of ISO 9001 software. In addition, the terminology and concepts of ISO/IEC 15939 are included in ISO/IEC 15288, ISO/IEC 9126 and ISO/IEC 14598.

The ISO/IEC 15939 standard was also used for the development of the "Measurement and Analysis (MA) Process Area of CMMI [Chr 03]. The Measurement and Analysis process area summarizes the activities needed to carry out the measurement process and provides a methodology for assessing whether the measurement process is carried out in accordance with the standard.

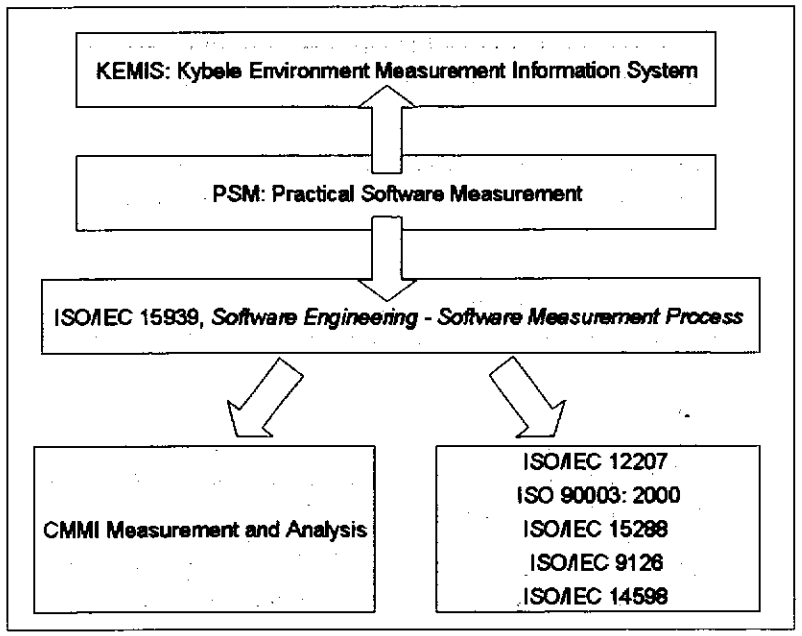


Fig. 1 *Measurement methodologies and standards*

The approach for selecting and specifying metrics proposed by PSM is based on the direct relationship between the information needs and the measures that support the information required.

For all these reasons, Kybele Consulting has identified PSM as a referral process for completing the definition of the KEMIS infrastructure, considering all those categories and metrics related to software quality and extending them with the prior knowledge and results from previous projects.

Using PSM and Maven 2 [Mas 07] measurement plug-ins, KEMIS proposes an adaptation which relates the metrics and indicators proposed by PSM to results obtained by the execution of plug-ins. Section 4 presents some of these metrics and indicators of product quality obtained by KEMIS.

2.2 Measurement Automation

To achieve the second and third goals (to measure regularly and frequently and in an automated way), KEMIS environment is based on Maven 2[2], a software tool for managing and understanding Java projects which through its measurement plug-ins allows us to get a set of metrics automatically. In addition, Maven 2 is based on the continuous integration concept [Fow 99] and, by using the Continuum[3] tool, it allows the scheduling of the measurement process.

Moreover, our experience in quality audits shows that the implementation of Maven 2, together with its measurement plug-ins, is ideal for conducting continuous improvement and periodic quality control of each project. Figure 2 shows that, when using Maven-based environments, the first time that project quality is assessed involves the greatest effort (execution in batch mode), as compared to other interactive environments.

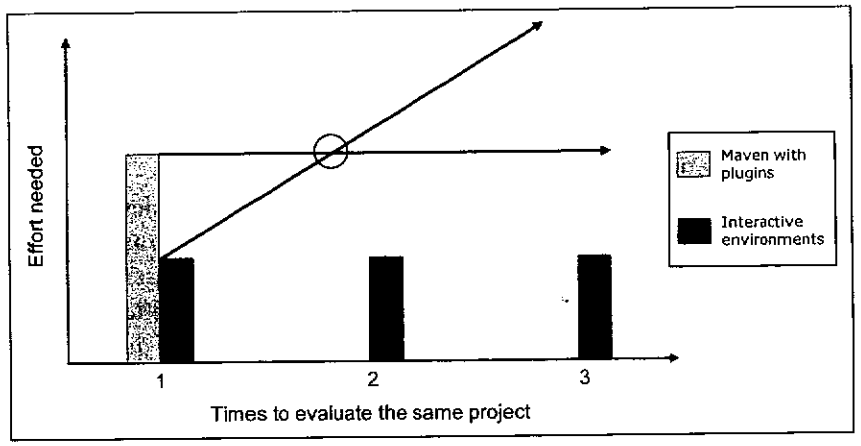


Fig. 2 *Evolution of Effort in assessing the quality of the same product*

However, while when using interactive environments the effort in the measurements following the initial one remains the same, the effort required when using Maven to do the same measurements is virtually zero. Results are obtained automatically, in spite of changes made in the project.

2.3 Display of Results

Once the quality measurement process has been automated, we must consider another threat: automation can generate so much information in a short time that sometimes this can be worse than not having information. That brings about

2 http://maven.apache.org/
3 http://maven.apache.org/continuum/

the need to have syntheses, historical, indicators, tendencies, a scale of information tools, etc. (this is sometimes known as a balanced scorecard).

For that reason, to fulfill the fourth objective (to obtain different abstraction levels in the presentation of results) and to make an intuitive analysis of software product quality possible, KEMIS proposes a set of reports, which gathers the main quality indicators. It uses a MySQL database where it stores the most representative results obtained by measurement plug-ins and a server in which the reports are registered, so we can later consult automatically and present the information on product quality.

3 Architecture of the KEMIS Project

Figure 3 shows KEMIS environment architecture. As can be seen at a glance, it emphasizes the dashed line that divides the figure. This separation corresponds to two phases of KEMIS implementation.

Basic measurement infrastructure (operative level): The first phase is the installation and configuration of Maven 2, as well as its measurement plug-ins. After this first phase, the user has an environment that allows him or her to get quality metrics in a way that is regular and automatic.

Advanced measurement infrastructure (tactical and strategic levels): The second phase relates to the environment installation and configuration for generation of reports. Upon completion of this second phase, the user has the complete KEMIS environment which, in addition to the activities of the first phase, obtains custom reports with key indicators of product quality.

The following steps summarize the operation of the global environment:

1. The regular execution of Maven 2 is planned by Continuum.
2. Maven 2 is responsible for an asynchronous execution of a predefined set of commands. These commands are configured in a file ("pom.xml").
3. Maven/Continuum are retrieved by means of the software configuration management tool (SCM) project files with which they must work.
4. Once files have been retrieved from SCM, Maven 2 performs the actions defined in its configuration.
5. Maven 2 plug-ins calculate the quality metrics about source files retrieved and store results in XML format.
6. The results are studied in order to extract the required information from them and to store it in the data base.
7. Meanwhile, quality report templates are created. These templates will show the main quality indicators.
8. The quality report templates are integrated and configured by means of a report server tool.

9. The Report server tool extracts from the data base the information required to create reports using SQL queries.
10. Finally, the report server tool generates reports with the data and format specified by the user.

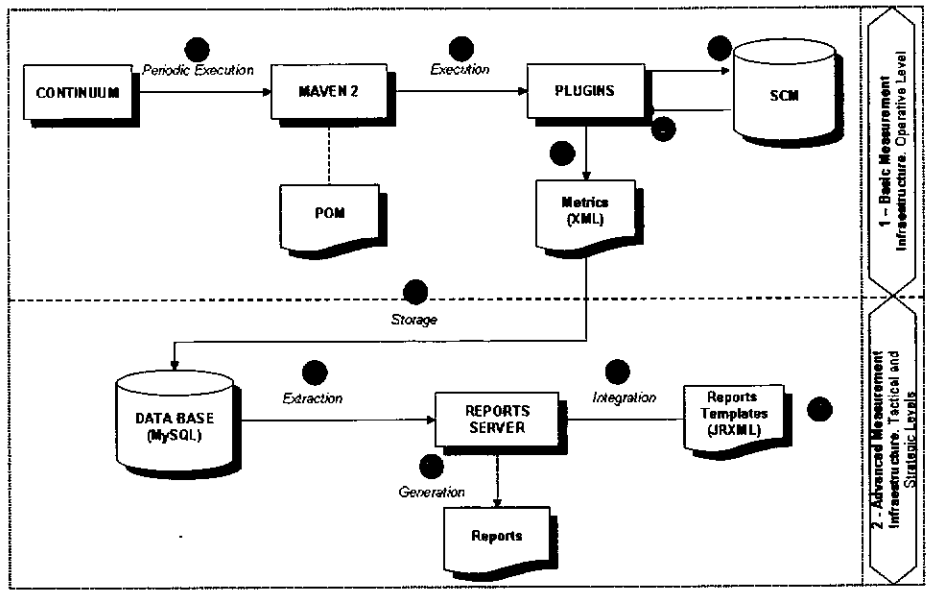


Fig. 3 Architecture of the KEMIS environment

4 KEMIS Metrics and Indicators

By means of Maven 2 measurement plug-ins, once the basic measurement infrastructure has been implanted, it is possible to obtain a comprehensive list of metrics.

4.1 Classification of Metrics

Table 1 presents the classification of the comprehensive list of metrics obtained. The classification has a set of categories, as well as some of the metrics belonging to these categories.

CATEGORY	METRIC IDENTIFICATOR
UNNECESSARY CODE	EmptyFinalizer (EF) FinalizeOnlyCallsSuperFinalize (FOCSF) UnusedPrivateField (UPF) EmptyIfStmt (EIS)
DOCUMENTATION	JavadocMethod (JDM) JavadocStyle (JDS) JavadocType (JDT)
PROGRAMMING STYLE	AssignmentInOperand (AIO) ForLoopShouldBeWhileLoop (FLSBWL) SwitchDensity (SD) VariableNamingConventions (VNC)
MAINTAINABILITY	ExcessiveImports (EI) AvoidDeeplyNestedIfStmts (ADNS) ExcessiveMethodLength (EML) ExcessiveParameterList (EPL)
POSSIBLE ERRORS	FinalizeOverloaded (FOL) AvoidCallingFinalize (ACF) NullAssignment (NA) EmptyCatchBlock (ECB)
PERFORMANCE	FinalFieldCouldBeStatic (FFCBS) OptimizableToArrayCall (OTAC) AvoidConcatenatingNonLiteralsInStringBuffer (ACNLISB) ExplicitInitialization (EI)
SOFTWARE QUALITY METRICS	Non Commenting Source Statements (NCSS) Javadoc lines (JDL) Cyclomatic Complexity Number (CCN) Total number of duplicate blocks (NDB) Cycles (C)

Tab. 1 *Categories and metrics obtained by means of measurement plug-ins*

4.2 Indicators

Due to the large amount of information generated by measurement plug-ins and the difficulty of managing such results as regards the way they are generated, KEMIS has proposed and provided a set of reports that cover the main quality indicators of the software product. These indicators are classified according to PSM into the following categories:

Components
Lines of Code
Code Defects
Cyclomatic Complexity
Duplicate Code
Software Quality

Next, to give an example of how all this works, we take a look at some of these indicators.

Code Defects

Indicators of this category are used to provide a global perspective on the code quality of the project under study in terms of the number of defects (being a defect the failure to comply with an established rule). In this category the following reports stand out:

Total defects in project source code. This indicator studies the total defects found in the code of a project and also shows the variation in defects that the project has undergone throughout the various measurements performed by the user.
Projects that are above a certain threshold of defects. This indicator studies those projects that exceed a certain number of total defects.
Changes in the density of defects detected in the code. This indicator studies the project density of defects, linking the total defects to the total lines of code (NCSS). Furthermore, it shows how the density of defects varies throughout the various measurements.
Projects whose density of defects is above or below a threshold. This indicator shows those projects which either do not meet or which exceed a certain threshold specified by the user with respect to the value of the density of defects.

Cyclomatic Complexity

Indicators of this category are used to provide a global perspective on the code quality of the project under study in terms of Cyclomatic Complexity. The following reports will be available to this category:

Changes in the project Cyclomatic Complexity. This indicator studies the project Cyclomatic Complexity. Furthermore, it also shows the variation in Cyclomatic Complexity that the project has undergone over the various measures made by the user.
Projects that do not meet or exceed a threshold value of total Cyclomatic Complexity. This indicator lets us know those projects that have a total Cyclomatic Complexity higher or lower than a specified limit.
Evolution of the average Cyclomatic Complexity of the functions of a particular project. This indicator studies the Cyclomatic Complexity average of a project function selected by the user. It also shows the evolution of this complexity over the various measurements.
Projects that do not meet or exceed a threshold value of average Cyclomatic Complexity per function. This indicator lets us know those projects which have a Cyclomatic Complexity average higher or lower according to a limit determined by the user.

Software Quality Indicator

This is the comprehensive indicator, whose function is to summarize the results of previous indicators, showing in a single report the information that is deemed most important in determining the quality of the projects studied. In Figure 4 we can see a Kiviat diagram with information about this indicator:

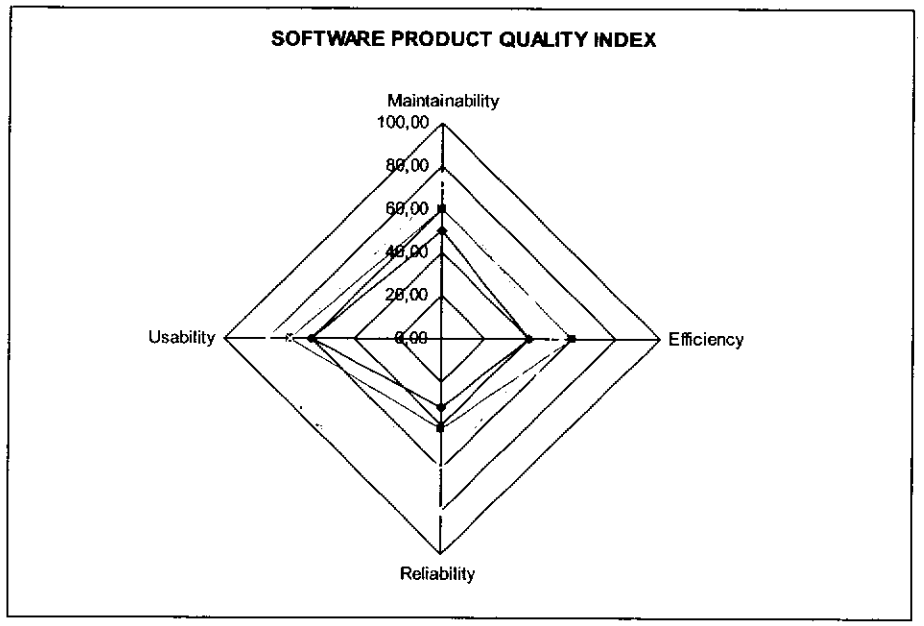


Fig. 4 *Software quality index*

5 KEMIS Contributions

Speaking in general terms, KEMIS can be used by an organization that wants to evaluate either the quality of software products that it receives, or those that have been developed internally. In addition, given the increasing need to gain certification in quality models like CMMI, companies need to implant a system of development process and software product measurement. KEMIS provides the organization with the infrastructure required to carry out product quality measurement.

KEMIS allows software factories to control the quality of the developments from the first stages, which is in turn translated into more efficient development, better final quality, and better economics. In addition, KEMIS is designed to be used by the developers, who can evaluate their work by means of the basic measurement infrastructure. For organizations that have outsourced their software development, KEMIS controls the quality of deliveries made by the suppliers,

making it possible to evaluate great amounts of developments before they go to production.

In Figure 5 there is a comparative table of the tools whose execution is done in batch mode, as over and against those presenting an interactive environment and those in a KEMIS environment.

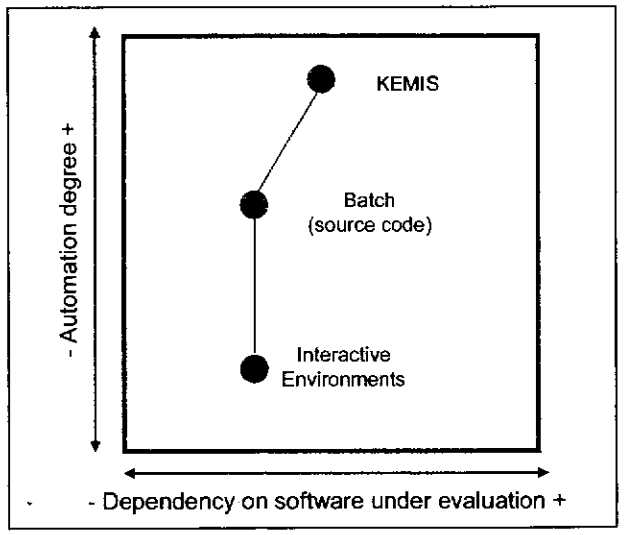


Fig. 5 *Comparative of quality measurement tools*

We consider the "archetype" tool to be one whose implementation would be completely automated and whose results are presented in the format desired by the end user. It would also be totally independent of the software being evaluated, with no need to be aware of it. All that would be necessary is to have the files with the code that is being evaluated. With this "archetype" we obtain the following conclusions:

The tools whose execution was carried out in batch mode have a high degree of automation, as they may be launched by a task scheduler, avoiding implicit user interaction when making the measurement. Interactive environments are just the opposite of all this, requiring supervision at all times.

The tools in batch mode use the source code as input, as do interactive environments (considering those which also work with the sources of the program). This means to have less software dependence, which reduces both problems and the time needed for measurement.

In the case of KEMIS, despite being somewhat more dependent on the software under evaluation (needing the libraries used in the project), it shows the highest degree of automation, because it conforms to the characteristics of the batch mode and also presents personalized results.

6 Conclusions and Future Work

In current software development, measurement is a core component in the checking of product quality, especially if external teams do the development. Here, the clients purchasing the software should establish their own quality control systems. It is at this point that the KEMIS project emerges as a solution for companies that want to measure the quality of their software products, especially for those that have outsourced the development of these products and who require an automated, regular and intuitive analysis of software product quality.

Below is a set of best practices and recommendations which we think are important, on the basis of our experience:

The quantitative product evaluation:

- It is a method that is necessary not only in development, but also if there is outsourcing, especially if we are responsible for the setting up of the production process.
- It allows us to identify symptoms, with an aim to implement correct solutions.
- It provides strong arguments for defending and raising awareness of the need to improve the process.

Sampling frequency is dictated by the complexity of the measurement system. The more expensive the measurement process is, the fewer the measurements will be and the lower the quality.

Today it is easy to collect a lot of data on software product quality, but the difficulty lies in knowing what to do with these.

You have to decide what you want to show, when and to whom. This is about abstraction and information levels.

The process of defining metrics is incremental and has to be improved continuously, according to the specific needs of the user and the product.

An activity that has been accepted very recently is the option of outsourcing product quality evaluation.

We should never lose sight of "the ultimate goal", which will be to align business objectives with improvement plans, as well as to increase productivity and develop profitability, increase software quality and user satisfaction, and so on.

Currently, there is a set of tasks planned for the future of the KEMIS project, among which we highlight the integration of new measurement and documentation plug-ins. This is an important issue to tackle, due to the increasing quantity of these and to the frequency with which the plug-ins are being developed today. An additional task is to store greater amounts of data from the measurement results, allowing us to define and generate new indicators and reports.

On the other hand, we are interested in extending the project to the Internet. This would be a step towards the outsourcing of product quality assessment, but the user would be able to have fully transparent access to the status of their projects in real-time.

Finally, we are working to extend KEMIS so it can be used not only with java projects, but also with other programming languages like C++, Visual Basic, Python, C#, etc.

References

[Won 04] B. Wong, et al.: Second Workshop on Software Quality. in 26th international conference on Software engineering. 2004. St. Louis, MO, USA: ACM Press

[Won 05] B. Wong, et al.: Third Workshop on Software Quality. in 27th international conference on Software engineering. 2005. St. Louis, MO, USA: ACM Press

[Fen 97] N. Fenton and S. Pfleeger, Software Metrics: A Rigorous Approach. 2nd edition, 1997, London, Chapman & Hall.

[Bri 96] L. Briand, S. Morasca, and V. Basili, Property-Based Software Engineering Measurement. IEEE Transactions on Software Engineering, 1996. 22(1): p. 68-86.

[Plf 97] S.L. Pfleeger, Assessing Software Measurement. IEEE Software, 1997. March/April: p. 25-26.

[Lav 00] L. Lavazza, Providing Automated Support for the GQM Measurement Process. IEEE Software 2000. 17(3): p. 56-62.

[Gil 95] A. Giles and G. Daich, Metrics Tools. Crosstalk, The Journal of Defense Software Engineering, 1995. February: p. http://www.stsc.hill.af.mil/crosstalk/frames.asp?uri=1995/02/Metrics.asp

[McG 06] J. McGarry, et al., Practical Software Measurement: Objective Information for Decision Makers. 2006.

[ISO 02] ISO/IEC, ISO/IEC 15939 International Standard Software engineering-Software measurement process. 2002.

[Chr 03] M.B. Chrissis, M. Konrad, S. Shrum, CMMI: Guidelines for Process Integration and Product Improvement, ed. T.S.S.i.S. Engineering. 2003: Addison Wesley Professional.

[Mas 07] V. Masol, et al., Better Builds with Maven, ed. DevZuz. 2007.

[Fow 99] M. Fowler, Refactoring. Improving the Design of Existing Code. 1999: Addison-Wesley.