

Waßmuth, Stefan ; Dambon, Martin and Linß, Gerhard:

***Development methods for quality measurement tools enabled by
Open Source Software***

Zuerst erschienen in:

11th QMOD Conference : quality management and organizational development attaining sustainability from organizational excellence to sustainable excellence, 20 -22 August, 2008 in Helsingborg, Sweden, 2008 / Su Mi Park Dahlgaard ... (eds.). - Linköping : Linköping Univ. Electronic Press, 2008. - (Linköping electronic conference proceedings ; 33)



Dieser Artikel ist ein Open Access-Artikel und steht unter den Creative Commons Lizenzbedingungen (<http://creativecommons.org/licenses/by-nc/3.0/deed.de>). Sie dürfen: den Inhalt vervielfältigen, verbreiten und öffentlich zugänglich machen, vorausgesetzt dass Autor und Quelle genannt werden.
Dieses Werk darf nicht für kommerzielle Zwecke verwendet werden.

Development Methods for Quality Measurement Tools enabled by Open Source Software

Stefan Waßmuth, stefan.wassmuth@tu-ilmenau.de

Martin Dambon, martin.dambon@tu-ilmenau.de

Gerhard Linß, gerhard.linss@tu-ilmenau.de

Technische Universität Ilmenau, Ilmenau, Germany

Department of Quality Assurance, Faculty of Mechanical Engineering

1. Abstract

The Free and Open Source Software Movement is currently the most interesting and influential trend in software industry as it enables nearly boundless access to software. On the other side future generations of engineers will need more and more profound software knowledge. The article describes a new education level for engineers in quality assurance developed and applied at the Technische Universität Ilmenau.

Through broad application of Open Source Software students without any previous knowledge are enabled to develop a customised multi-layer software system to specific problems of practical relevance. The software system is based on an Office client whether from Microsoft Office or OpenOffice connected with a SQL database server. The paper details which difficulties occurred most often during development and to what extent the developed software solutions could be deployed in industry as a cost efficient alternative.

2. Purpose

University education has its strength in teaching theory and methodical understanding. Nevertheless practical relevance is sometimes absent or acquired knowledge could not be adopted adequately and therefore is forgotten promptly. On the other hand it could be ascertained that most students still have deficiency in software knowledge. That implies the capability to describe a problem in an abstract manner in order to be later implemented easier into software as well as to be well versed with a sort of software tools which are considered as elementary. The necessary knowledge of a certain programming language can thereby differ from case to case. Both Office suites (whether form Microsoft or OpenOffice.org) and SQL databases are considered as very useful tools by the authors of this article in the context of quality assurance.

Microsoft Office (MS Office) is maybe the most distributed commercial software package around the world as it could be widely used and customised by the user himself. OpenOffice (full: OpenOffice.org) is a serious alternative to MS Office as it is Open Source and for free (see “Excursus: Free and Open Source Software” below).

SQL databases are the most commonly used type of databases. A database is an important element of every bigger software information system as it provides a simultaneous access and the possibility of storing and retrieving huge quantities of data.

The lack of software knowledge in industry today is shown in a variety of international surveys (Buschermöhle, 2006; Standish 2004; Sauer 2003). Accordingly, most software projects in enterprises are failing. Either they are cancelled or they are completed at the cost of budget or time overrun or with a reduced functional scope, fig. 1.

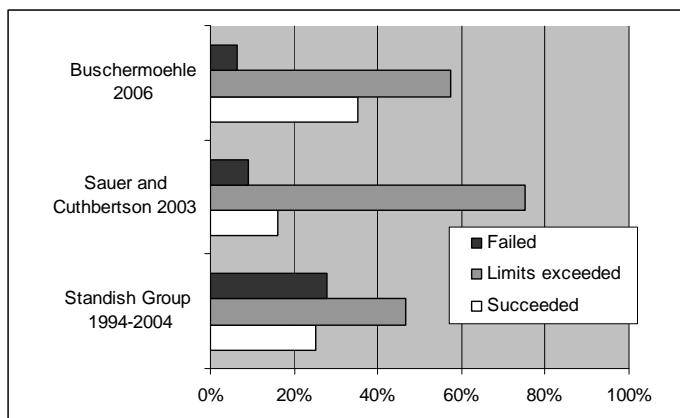


Fig. 1 – Outcomes of surveys investigating software projects in enterprises (Buschermöhle, 2006; Standish, 2004; Sauer 2003)

Nowadays there are different alternatives to consider for software development: in-house development, outsourcing/ Commercial off-the-shelf (COTS) procurement and Free/ Open Source Software (F/OSS). Studies on the application of software for quality assurance in Germany show that only one third of all users purchase Commercial off-the-shelf Software for quality assurance (Waßmuth, 2008; Roßdeutscher, 2007; Krautwasser 2005). Most companies either disclaim the application of any specialised software or develop it by their own (in-house development). The last ones are often based on MS Office (Roßdeutscher, 2007).

Referring to the mentioned deficits in education and industry as well, the British Computer Society (BCS) states: “Improving education will not make an immediate change to practice, but is a vital part of a long-term solution to the problem...” (BCS, 2004).

Therefore a new approach in education has been developed and applied at the Technische Universität Ilmenau. Based on a SQL database server and Office clients students in mechanical engineering are enabled to develop customised software solutions to problems of practical relevance without having any specific knowledge in advance. Applied Open Source Software tools made development process easier and software solutions more powerful on the other hand. The following objectives should be accomplished:

- Enhance knowledge in quality assurance through practical experience,
- Gaining understanding about the process of software development and architecture of multi-layer software systems,
- Learning to work with useful software for later job,
- Training of presentation and teamwork.

From the scientific point of view, the frequent problems at the development of software and the evaluation of quality of software solutions were of particular importance.

Excusus: Free and Open Source Software

The original definition of “Free Software” comes from the Free Software Foundation and it asserts, that a program is free, if the user has the freedom to run, copy, distribute, study, change and improve the software (FSD, 2008). The last ones require access to source code as a precondition. Although the terminology of “free” should be interpreted originally for “freedom” instead of “not to pay”, usually it is possible to obtain so referred software without paying money and simply downloading it from the Internet (Balduzzi, 2005).

The “Open Source” definition, on the other hand is quite similar. The Open Source Initiative describes it as a development method for software that harnesses the power of distributed peer review and transparency of process (OSD, 2008).

The two terms “Free Software” and “Open Source Software” refer to two separate movements, Open Source is a development methodology and Free Software is a social movement. In fact both terms are used to indicate the same kind of software referred to as Free and Open Source Software (F/OSS) (Balduzzi, 2005). F/OSS has some advantages in comparison to “normal” software as well as some weak points, too. The most mentioned Pro’s and Con’s are summarized in Table I.

Pro's	Con's
lower costs	different stage of maturity
reliability	no guarantee
interoperability	software licences
independency	

Table I – Pro’s and Con’s of F/OSS (Wheeler, 2008; Balduzzi, 2005; Wieland, 2004)

Firstly F/OSS guarantees lower TCO (Total Cost of Ownership), because there are no licensing fees and users are not tied to monopolistic vendors. F/OSS promises higher reliability, as more people are involved in software development, source code review and usage. F/OSS allows afterwards an easy connection between information systems, because of its access to source code and used standards definitions. Users can re-use and adapt existing open source implementations in other systems. Users do not depend on proprietary providers since other providers have access to the same knowledge and technology (Wheeler, 2008; Balduzzi, 2005).

On the other side there are partly huge discrepancies at the stage of maturity between projects. As F/OSS can be simply downloaded from the Internet for free there is no guarantee on the software and the possibility that software is no longer developed. Furthermore there are different software licenses – some of them have nothing in common with the original definition of “Free” or “Open Source Software”. Therefore possible software candidates should be analysed closely. A huge community is always a good indicator as well as high download rates (Wieland, 2004).

Today more and more enterprises and governments starting to adopt F/OSS instead of COTS, for F/OSS usage in governments see fig. 2.

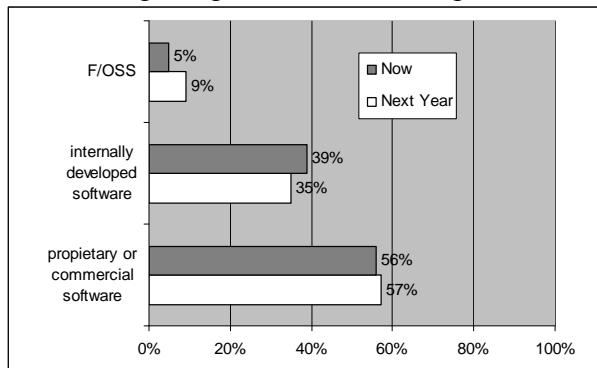


Fig. 2 – F/OSS in Government: Today vs. Next Year; survey found in (Rosenthal, 2006)

Some major companies, e.g. IBM and Sun Microsystems have joined some F/OSS projects and support them. A lot of established businesses provide services around F/OSS for-fee, e.g. consulting, customising or enhancements. The success of the new software movement is shown by many case studies, some very popular are: the Linux operating system, the Apache web server, the Firefox web-browser, the Eclipse development framework, the MySQL database server and OpenOffice as a serious alternative to other Office suites. Huge Internet repositories like www.sourceforge.org and www.freshmeat.net show success of the movement with thousands of registered projects. More information on F/OSS, important projects, different license models etc. can be found on e.g. (Wheeler, 2008).

3. Approach

The extent of selected F/OSS for education purpose is demonstrated in fig. 3 and shows a viable software architecture which also could be deployed in industry as a cost efficient alternative to COTS solutions.

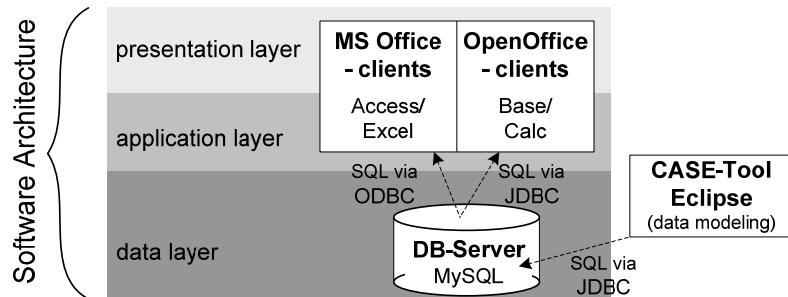


Fig. 3 – Selected software components

The chosen components except Microsoft Office client – are popular F/OSS software: an Eclipse plug-in (Eclipse, 2008) as CASE-tool (Computer Aided Software Engineering) for data modelling, MySQL (MySQL, 2008) as database system and the OpenOffice client (OpenOffice, 2008). Connection between the software components had been made through SQL via JDBC and ODBC middleware.

To build up a complex software system it is necessary to apply a systematic approach. The software lifecycle could be divided into at least four main steps: requirements analysis, design, implementation and maintenance. Using an adequate modelling language, tools can help to verify the underlying model as well as to generate executable code.

In practice data models are of an outstanding importance at the development of complex software systems. Data are the core of every software system. Data do even exist without a certain program and therefore have a higher stability than the program itself. Although the data model is just one part of system development, it has a strong impact on the later software, i.e. user requirements, application integration and costs (Ponniah, 2007). A data model provides a method for describing the real-world information requirements in a manner understandable to all stakeholders. It also serves as a blueprint of the database system for the developers. The adopted approach to create an Office - database centred software system is shown in fig. 4. All main steps of software lifecycle mentioned above are included.

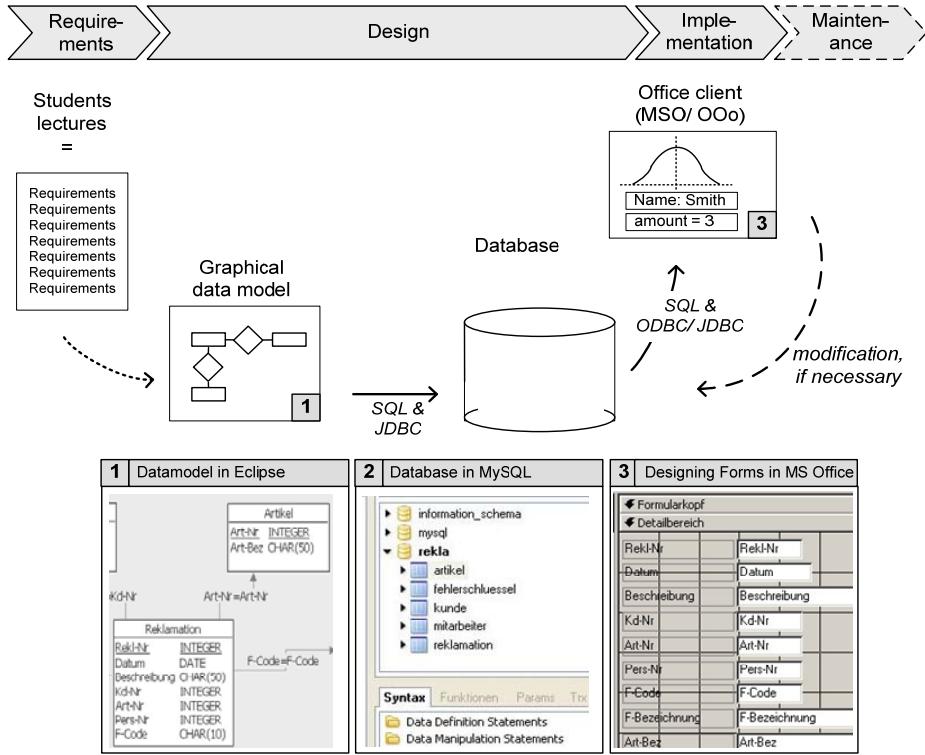


Fig. 4: Development process and applied software components

Every group was made up of two or three students and had to select a typical field of interest in quality assurance to be later realised with software. At the beginning each student had to give a lecture which should include the theoretical basics as well as a plausible example to the other students. The lecture was also seen as the requirements document and the result of requirements analysis.

4. Findings

With one exception all groups had shown viable Office clients using Microsoft Excel/ OpenOffice Calc and/or Microsoft Access/ OpenOffice Base. (Note: Initially only Microsoft Office package had been used. From the second semester on also OpenOffice.org had been applied.)

Typical software tools which had been realized are, among others:

- Machine and Process Capability Analysis,
- Statistical Process Control (Control Charts),
- Acceptable Quality Level (AQL) Sampling Inspection,
- Supplier Assessment,
- Complaints Management,
- Failure Mode and Effects Analysis (FMEA),
- ...

Relevant data as master data and measuring data could be entered into the program by manual input, by data file like *.csv or even through connection to database, using SQL-commands. To following both examples should an overall impression of realised software solutions

Example 1 – Supplier Assessment

Fig. 5 shows the students data model for supplier assessment as domain of interest.

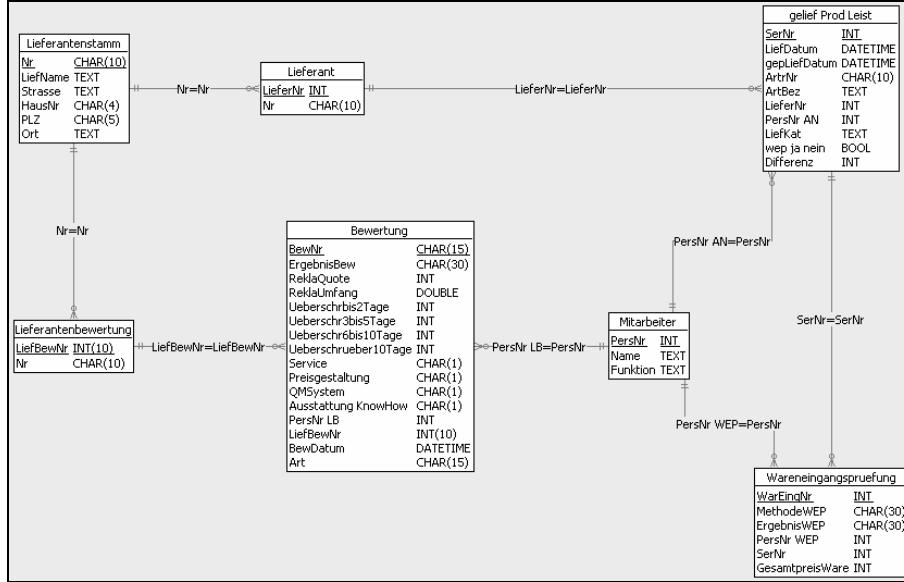


Fig. 5: Students data model “Supplier Assessment”, made with (Eclipse, 2008)

The program covers elementary functionality for supplier assessment: management of necessary master data like supplier and material, the handling of goods inspection and the integration toward supplier assessment, fig. 6. The management of supplier assessment includes the calculation of hard facts as delivery reliability or error rates as well as soft facts like benchmarking the overall service quality or the quality management system.

The screenshot displays two side-by-side windows of the 'Supplier Assessment' application:

- Left Side (Microsoft Office):**
 - Lieferantenbewertung** window: Shows input fields for Reklamationsquote (14%), Reklamationsumfang (150 T€), Überschreitungen bis 2 Tage (1), Überschreitung 3 bis 5 Tage (1), Überschreitung 6 bis 10 Tage (1), Überschreitung über 10 Tage (2), Service (1), Preisgestaltung (2), QM-System (2), Ausstattung/Know How (1), Ergebnis (216), and Ergebnis in Worten (C-Lieferant). A legend at the bottom indicates: 1 - sehr gut, 5 - unbefriedigend.
 - Bewertung von Lieferanten** window: Shows a table with columns Kriterium, Faktor, Punkte, and Bemerkung. Data rows include: Reklamationsquote in %: 2, 33; Rücksichtnahme lang in T€: 1, 0,064; Überschreitung < 2 Tage: 1, 1; Überschreitung 3 bis 5 Tage: 2, 4; Überschreitung 6 bis 10 Tage: 3, 3.
- Right Side (OpenOffice):**
 - Lieferantenbewertung** window: Shows a table with columns Kriterium, Faktor, Punkte, and Bemerkung. Data rows include: Reklamationsquote in %: 2, 33; Rücksichtnahme lang in T€: 1, 0,064; Überschreitung < 2 Tage: 1, 1; Überschreitung 3 bis 5 Tage: 2, 4; Überschreitung 6 bis 10 Tage: 3, 3.
 - Lieferkategorie** window: Shows a radio button group for Produkt (selected) and Dienstleistung.
 - Wareneingangsprüfung** window: Shows a table with columns WEP-Nr., KdNr, KdName, Bearbeiter, Prüfmethode, Warenpreis, and Ergebnis WEP. Data rows include: WEP-Nr.: 2, KdNr: 1, KdName: Bosch, Bearbeiter: Herbert Sausmann, Prüfmethode: genau hingeklickt, Warenpreis: 1, Ergebnis WEP: 1, Reaktion: Keine Reaktion.
 - Personal-Nr. Annahme** window: Shows a table with columns PersNr, Name, and Annahme. Data rows include: PersNr: 1, Name: Bosch, Annahme: 1; PersNr: 2, Name: Dodl Huster, Annahme: 1; PersNr: 3, Name: Irmgard Schmidt, Annahme: 1; PersNr: 4, Name: Puk Kochrowski, Annahme: 1; PersNr: 5, Name: Hannes Driessens, Annahme: 1; PersNr: 6, Name: Wolfgang Anselm, Annahme: 1.
 - Datensatz** window: Shows a table with columns PersNr, Name, Funktion, and Annahme. Data rows include: PersNr: 1, Name: Bosch, Funktion: Lieferant, Annahme: 1; PersNr: 2, Name: Dodl Huster, Funktion: LB, Annahme: 1; PersNr: 3, Name: Irmgard Schmidt, Funktion: Administ, Annahme: 1; PersNr: 4, Name: Puk Kochrowski, Funktion: Annahme, Annahme: 1; PersNr: 5, Name: Hannes Driessens, Funktion: WEP, Annahme: 1; PersNr: 6, Name: Wolfgang Anselm, Funktion: Annahme, Annahme: 1.
 - Summe** window: Shows a table with columns PersNr, Name, Funktion, and Annahme. Data rows include: PersNr: 1, Name: Bosch, Funktion: Lieferant, Annahme: 1; PersNr: 2, Name: Dodl Huster, Funktion: LB, Annahme: 1; PersNr: 3, Name: Irmgard Schmidt, Funktion: Administ, Annahme: 1; PersNr: 4, Name: Puk Kochrowski, Funktion: Annahme, Annahme: 1; PersNr: 5, Name: Hannes Driessens, Funktion: WEP, Annahme: 1; PersNr: 6, Name: Wolfgang Anselm, Funktion: Annahme, Annahme: 1.

Fig. 6: “Supplier Assessment”, left side: Microsoft Office, right side: OpenOffice

Example 2 – Statistical Process Control

Purpose of this Statistical Process Control is monitoring a process routinely through the use of attribute control charts. The first picture shows underlying data model supporting process specific data and measuring data of certain characteristics, fig. 7.

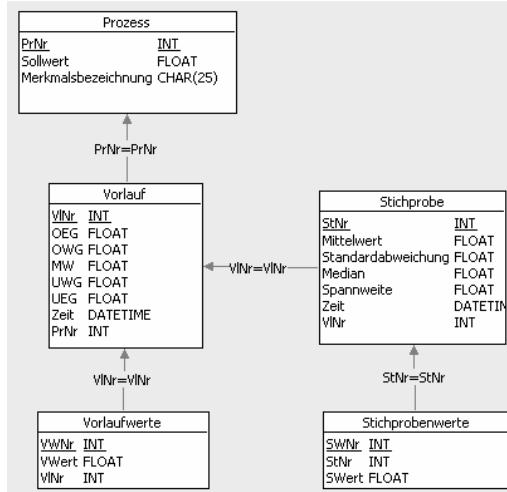


Fig. 7: Students data model “Statistical Process Control”, made with (Eclipse, 2008)

The software solution allows data input through forms in Microsoft Access or OpenOffice Base. Data are equally stored in the MySQL database. Measured data could be retrieved from the database or imported via *.csv file. The calculation was realised within Microsoft Excel or OpenOffice Calc. Both programs calculate corresponding upper/ lower control and action limit and plot the chart automatically, fig. 8.

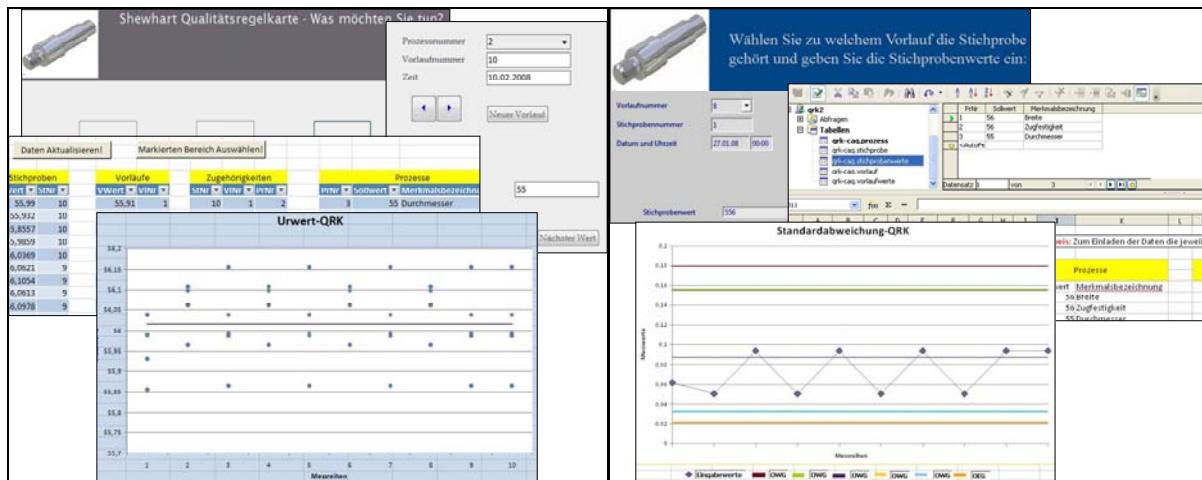


Fig. 8: “Statistical Process Control”, left side: Microsoft Office, right side: OpenOffice

Of the 68 students who attended the courses during that two semesters from the beginning only 47 passed it successfully. Those who changed (all within the first three weeks) reclaimed the additional effort regarding to a “normal” course. At the beginning and at the end of the semester a standardised questionnaire was handed out to each student. One intention was to evaluate student’s knowledge concerning Microsoft Office/ Open Office and databases, both through self assessment and control questions. Fig. 9 shows a good correlation and an overall remarkable increase in knowledge.

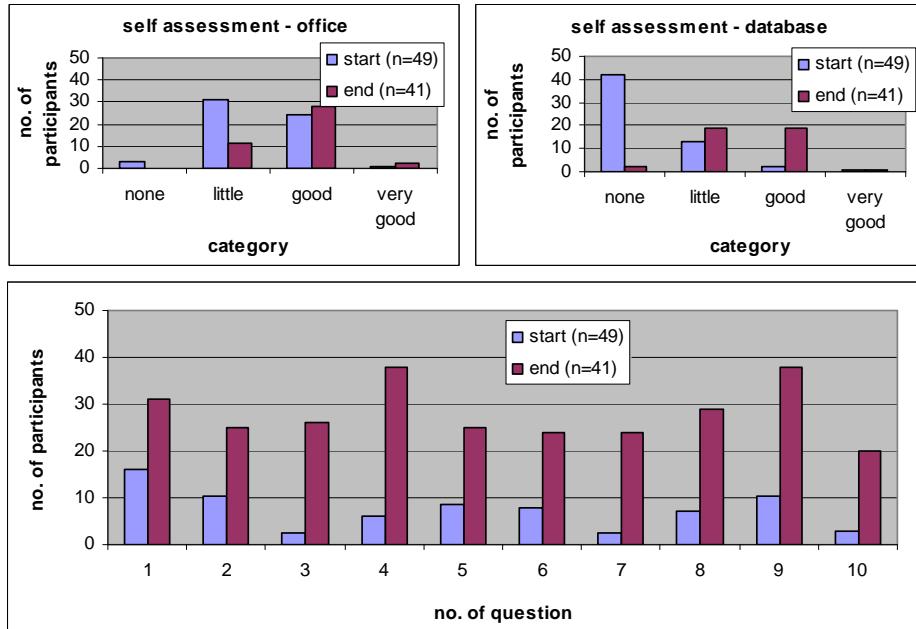


Fig. 9: Increased knowledge, above self assessment – below control questions (n=41, note: six forms were not applicable)

Another matter of interest was to find out what caused the most problems: creating the database or programming the Office client. For that purpose ten points had to be allocated. The number 10 stands for “most difficult”. Fig. 10 illustrates the results as Box-Whisker Plot. According to this students had more problems with Office than with the database.

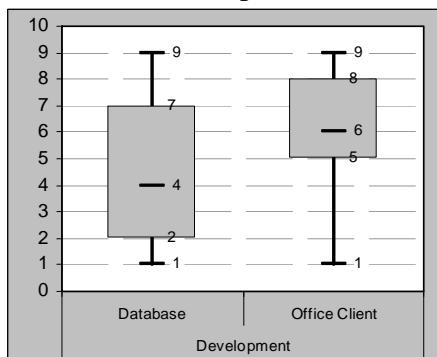


Fig. 10: Problems at the development: 10 – difficult, 0 – easy (n=41)

From the second semester on students should additionally develop a second client version with OpenOffice. To compare both clients students had to fill out a questionnaire to evaluate differences between Microsoft Office and OpenOffice. For that purpose 10 points had to be allocated among the two Office suites. A high number stand for higher quality. For evaluation of software quality four criteria had been chosen: Usability, Reliability, Efficiency and Functionality. These four criteria are part of the international standard ISO 9126 (ISO9126, 2001) used for classification of software quality:

- Usability measures the effort needed for use by a set of users.
- Reliability measures the capability of software to maintain its level of performance under stated conditions for a stated period of time.
- Efficiency measures the relationship between the level of performance of the software and the amount of resources used under stated conditions.
- Functionality measures existence of a set of functions and their specified properties. The functions are those that satisfy.

The standard additionally mentions Maintainability and Portability which emerge mainly in the last step of software lifecycle and therefore could not be evaluated by students so far.

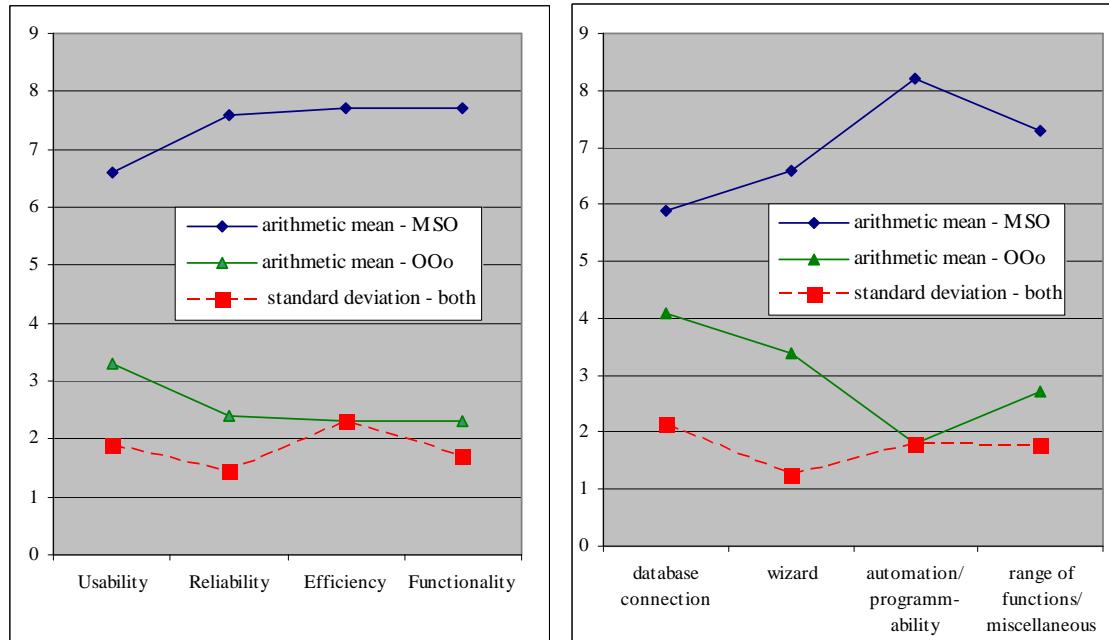


Fig. 11: Evaluation of both Office suites regarding specific quality criteria (left side) and commonly used functions (right side), n=14

Fig. 11, left side, shows assessment between Microsoft Office and OpenOffice through quality criteria. Microsoft still beats OpenOffice beyond all criteria. Although regarding “Usability” Microsoft Office and OpenOffice are quite close to each other. Similar results show the right picture of Fig. 11. Additionally to the four criteria students had to assess typical functions in Office accompanying the development process. Regarding to this OpenOffice still has some graver weak points in programmability and therefore automation whereas installing the connection to database server (here: MySQL) worked very well on both Office suites. Although most students had indicated fewer problems with the database (fig.10) it was evident that in most cases data models could only be applied to that special use case for which it was planned. By using the relational data model and suitable database design tools, building a database is not that difficult in fact. The problem lies more in the modelling effort or rather the development of models of higher quality (Simsion, 2005), fig. 12.

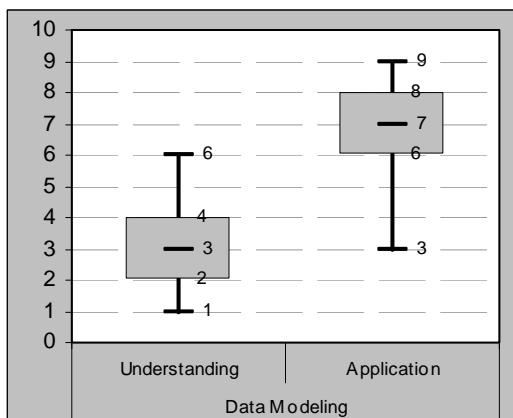


Fig. 12: Problems in data modelling: 10 – difficult, 0 – easy (n=41)

In principle there is always more than one solution to a certain (modelling) problem. The quality of a data model refers to understanding and evaluating the data model regarding customer's expectations (Shanks, 1997). In literature a lot of criteria frameworks have been proposed which can be used to understand and evaluate quality of data models (Shanks, 1997; Genero, 2002; Patig, 2006; Calero, 2002). Those quality criteria which are frequently mentioned and most important are:

- Correctness: A model should conform to the syntax rules of the particular notation.
- Completeness: Does a model include all necessary information? What can be expressed?
- Flexibility: Changing requirements should not cause changes to the data model or at least no drastic changes.
- Understandability: It can be achieved through simplicity and naturalness.

Some empirical studies on data modelling (Hitchman, 1995; Shanks, 1997; Castro, 2005; Leung, 2005; Simsion, 2005) have shown that:

- Data modelling novices didn't have problems using simple semantic constructs. Difficulties had increased by dealing with complex constructs and on a larger domain of interest.
- High quality data models can be achieved by a systematic approach. Experts are able to create better models due to their ability of abstraction, they use patterns of former projects and spend more time to review their models than novices do.

Regarding to fig. 10 only few students weighted "database development" higher respectively more difficult. It may be assumed that those students put more effort to adjust their data model and so created better models, see fig. 13.

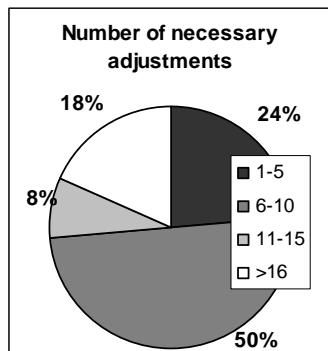


Fig. 13: Number of adjustments to data model (n=41)

5. Originality

The paper presents a systematic approach for development of customised quality measurement tools. The new approach had been tested with students at the Technische Universität Ilmenau. Because this method could be applied so easily and is so effective as well, it has now become an integral part of education in quality assurance at the Technische Universität Ilmenau. As it had been outlined, students were able:

- To create small individualized software applications for quality assurance using Office clients and a SQL database system,
- Without any previous software knowledge (fig. 10),
- And to develop and customise it by their own in a relatively short time – a necessary attribute for effective working with software.

During development every group became well aware of all those problems and necessary steps which have to be overcome and are so typical in every major software project. This training approach had been enabled only through broad application of F/OSS software tools, rapid testing and customizing as well as through good software documentation on the Internet and program assistance within the tools.

Most developed software solutions were limited to the initially proposed course examples instead of covering a more general application range. This was mainly due to limitations of the underlying data model. Developing high quality data models needs both comprehensive expertise and data modelling understanding. Nevertheless the majority of all participants were very enthusiastic regarding the new approach in education compared to standard courses, fig. 14. Interestingly nearly one half of the students were quite sceptical at the beginning of the course.

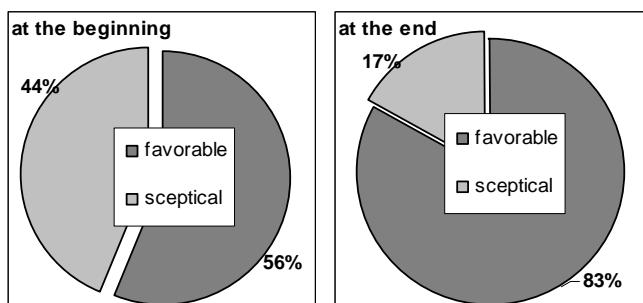


Fig. 14: Attitude to course at the beginning and at the end (n=41)

Keywords: education in quality assurance, database and Office-client, software development

Paper type: Case study

List of references:

- (Balduzzi, 2005) Balduzzi, M.: On the Influence of Free Software on Code Reuse in Software Development <http://www.idi.ntnu.no/emner/dt8100/Essay2005/balduzzi.pdf> [08-05-30]
- (BCS, 2004) Report of a working group from The Royal Academy of Engineering and The British Computer Society: The Challenges of Complex IT Projects, 2004, <http://www.bcs.org/upload/pdf/complexity.pdf> [08-05-30]
- (Buschermöhle, 2006) Buschermöhle, R.; u. a.: Success – Erfolgs- und Misserfolgsfaktoren bei der Durchführung von Hard- und Softwareentwicklungsprojekten, 2006, <http://vsek01.informatik.uni-oldenburg.de/~joomla> [08-05-30]
- (Calero, 2002) Calero, C.; Piatini, M.: Metrics for databases - a way to assure the quality. In: Information and Database Quality. Boston: Kluwer Academic Publ., 2002
- (Castro, 2005) Castro, E.; et al.: An empirical perspective of using ternary relationships in database conceptual modelling. In: Informatics in Education 2, V. 2, p. 191-200, 2005
- (Eclipse, 2008) Homepage of the Eclipse Project and the Azzurri Data Modelling Plug-in www.eclipse.org and www.azzurri.jp/en/software/clay/ [08-05-30]
- (FSD, 2008) The Free Software Definition, <http://www.gnu.org/philosophy/freesw.html> [08-05-30]
- (Genero, 2002) Genero, M. F.; Piattini, M. G.: Conceptual Model Quality. In: Information and Database Quality. Boston: Kluwer Academic Publ., 2002
- (Hitchman, 1995) Hitchman, S.: Practitioner perceptions on the use of some Semantic Concepts in the Entity Relationship Model, European Journal of Information Systems, 1995
- (ISO9126, 2001) International Organization for Standardization: Software engineering: Product quality, Genève: International Organization for Standardization, 2001 (International standard ISO/IEC : 9126)
- (Krauswasser, 2005) Fuchs, J.; Krautwasser, St.: Falsch geschätzt – Trendstudie: Wie viel Software braucht das Qualitätsmanagement? In: Qualität und Zuverlässigkeit QZ, München: Hanser, 50, p.20-23, 2005
- (Leung, 2005) Leung, F.; Bolloju, N.: Analyzing the Quality of Domain Models Developed by Novice Systems Analysts. In: Proceedings of the 38th Annual Hawaii International Conference on System Sciences 7, Nr. 7, p. 7, 2005

- (MySQL, 2008) Homepage of MySQL www.mysql.com [08-05-30]
- (OpenOffice, 2008) Homepage of OpenOffice.org <http://www.openoffice.org> [08-05-30]
- (OSD, 2008) The Open Source Definition, <http://www.opensource.org/docs/osd> [08-05-30]
- (Patig, 2006) Patig, S.: Die Evolution von Modellierungssprachen. Berlin: Frank & Timme, 2006 (habilitation)
- (Ponniah, 2007) Ponniah, P.: Data modelling fundamentals - a practical guide for IT professionals, Hoboken: Wiley: 2007
- (Rosenthal, 2006) Rosenthal, M.; Murray Rosenthal, CISA City of Toronto
<http://www.myhamilton.ca/NR/rdonlyres/149C4042-1B16-4B5E-91AB-1CFAC93F1335/0/MurrayRosenthalWellBehavedSystems.pdf> [08-05-30]
- (Roßdeutscher, 2007) Roßdeutscher, A.: Untersuchungen zum CAQ-Einsatz in KMU, TU Ilmenau, 2007 (diploma thesis)
- (Sauer, 2003) Sauer, C.; Cuthbertson, C.: The State of IT Project Management in the UK, 2003, <http://www.bestpracticehelp.com/The%20State%20of%20IT%20Project%20Management%20in%20the%20UK%202003-2004.pdf> [2008-05-30]
- (Shanks, 1997) Shanks, G.: Conceptual Data Modelling – An Empirical Study of Expert and Novice Data Modellers. In: Australasian Journal of Information Systems 4, Nr. 2, p. 63-73, 1997
- (Simsion, 2005) Simsion, G.: Understanding Data Model Quality, The Data Administration Newsletter 2005 <http://www.tdan.com/print/5100> [08-05-30]
- (Standish, 2004) The Standish Group International, 1994-2004,
<http://www.standishgroup.com/index.php> [08-05-30]
- (Waßmuth, 2008) Waßmuth, St.; Dambon, M.; Linß, G.: Anwendungsentwicklung für den CAQ-Bereich – Ergebnisse eines neuartigen Ausbildungskonzeptes an der TU Ilmenau. In: Innovationsqualität: Qualitätsmanagement für Innovationen: Bericht zur GQW-Tagung 2008 - Bremen / Gert Goch (Hrsg.) - Aachen: Shaker, 2008, p.79-101
- (Wheeler, 2008) David A. Wheeler: Why Open Source Software / Free Software (OSS/FS)? Look at the Numbers!
www.dwheeler.com/oss_fs_why.html [08-05-30]
- (Wieland, 2004) Wieland, T.: Stärken und Schwächen freier und Open Source Software im Unternehmen http://www4.informatik.tu-muenchen.de/lehre/vorlesungen/vse/WS2004/OSS_staecken_schwaechen.pdf [08-05-30]