Software Quality Modeling Experiences at an Oil Company

Constanza Lampasona & Jens Heidrich
Fraunhofer IESE
Fraunhofer-Platz 1
67663 Kaiserslautern, Germany
{constanza.lampasona | jens.heidrich}@iese.fraunhofer.de

Victor Basili
University of Maryland,
Fraunhofer CESE, and
King Abdulaziz University
825 University Research Court Suite
1300, College Park, MD, USA
basili@fc-md.umd.edu

Alexis Ocampo
ECOPETROL S.A.
Calle 37 No 7-43
Bogotá, Colombia
alexis.ocampo@ecopetrol.com.co

ABSTRACT
The concept of "software quality" is often hard to capture for an organization. Quality models aim at making the concept more operational by refining the "quality" of software development products and processes into sub-concepts down to the level of concrete metrics and indicators. In practice, it is difficult for an organization to come up with a reliable quality model because quality depends on numerous organizational context factors, and the model as well as the metrics and indicators need to be tailored to the specifics of the organization. This paper presents experiences in developing custom-tailored quality models for an organization, exemplified by Ecopetrol, a Colombian oil and gas company. The general approach taken is illustrated and excerpts from the initial model are presented.

Categories and Subject Descriptors
D.2.8 [**Software Engineering**]: Metrics – *Product metrics*

General Terms
Measurement

Keywords
Quality measurement and assurance, Industrial experience.

1. INTRODUCTION
Quality is defined as "the totality of characteristics of an entity that bear on its ability to satisfy stated and implied needs" (ISO 8402). Typically, the term "software quality" is hard to capture for an organization. In order to make the concept more operational, software quality models have been defined. These models try to refine the abstract term "quality" of software development products and processes into sub-concepts down to the level of metrics and indicators to allow quality measurement. One of the most popular product quality models is ISO 9126 and its successor ISO 25000.

When applying these models in practice, the problems are that (1) there exist a huge number of different quality models for different purposes, (2) the abstraction level is too high for applying the model, and most of all, (3) it is very difficult to come up with reliable metrics and indicators for an organization. One of the major reasons is that quality depends on stakeholders, application context, usage purpose, and other context factors of an organization and the models as well as the metrics and indicators need to be tailored to the specifics of the organization.

ESEM'12, September 19–20, 2012, Lund, Sweden.

This paper describes first steps towards creating a quality model for the IT department of Ecopetrol, a Colombian oil and gas company. One major goal of Ecopetrol is to become one of the 30 most important companies in the oil and gas domain by 2020. IT plays an important role in supporting the business processes necessary to establish Ecopetrol's top-level goals. Equally, it is important to align the incorporation, acquisition, and development of IT solutions with the company's landscape so that the goals achieved by IT directly impact the business goals. The company supports standardization of business processes and IT based on the premise that standardization will help it to expand and grow faster and be more agile. The paper presents the ongoing work of defining a quality model for the software that is acquired, maintained, or developed by external suppliers of Ecopetrol's IT department. The major goal related to developing the quality model was to improve software quality, reduce the issues caused by unknown/probably poor software quality, and in turn contribute to the ability to develop and maintain software faster and support Ecopetrol in becoming more agile. The quality model will support the definition of baselines for software quality at Ecopetrol and will become part of its IT landscape. It will help Ecopetrol define policies to be followed by the organization in order to achieve the goal of standardization of quality requirements.

This paper presents our practices and experience in developing quality models for an organization's specific needs. Section 2 presents related work in the field of quality modeling and the problems with the application of standardized quality models. Section 3 presents the approach taken to develop the model and introduces an excerpt of the developed model. The approach consisted of a survey of key stakeholders, a highly interactive GQM workshop, and the extraction of the model based upon the identification of problems and potential mitigation strategies. This led to the definition of the local meaning of quality and set the stage for evaluating the product based upon that definition. Suggestions for mechanisms for improving quality within the organization were also derived. Finally, Section 4 presents a short summary and conclusion and highlights future work aimed at completing the model and introducing the model into the organization.

2. RELATED WORK
Many quality models specify a prescriptive set of quality characteristics or metrics. ISO 25010 [8], for example, provides a structure for defining a set of quality attributes and sub-attributes. However, the direct use of this model is difficult because it is not easy to operationalize, which is an issue we also observed for other well-known models such as [4], [7], and [9]. Nevertheless, they provide a useful reference for defining high-level quality attributes.

Besides these general focus models, there exist many models explicitly developed for specific domains. Some examples are [2], which presents a maintainability model for embedded systems; [5], which presents a model for object-oriented design; and [6], which is a quality model specialized for European aerospace projects.

Kläs et al. [10] studied approaches for tailoring quality models and offer an approach for adapting existing quality models. We could not use the approach because we did not have an existing, worked-out model; Ecopetrol decided to only use an excerpt of the ISO25010 standard and develop their metrics from scratch.

We found only one model for measuring the quality of software in the oil industry [11]. Although the model's domain coincides with our domain of interest, we did not use it because it focuses only on the evaluation of discrete-event simulation software.

Because of the flexibility provided by the GQM paradigm [1], which allows for specifying a customer-tailored measurement program connecting high-level measurement goals and metrics, we chose to apply it to derive a quality model for Ecopetrol. The main reason for applying the GQM paradigm was its ability to provide a model that connects high-level quality aspects (as found in common standards, such as maintainability or reliability) with concrete metrics found on the operational level of the organization. Further advantages of applying GQM are that it can be used for (1) evaluating all artifacts from all development phases; 2) identifying strengths and weaknesses of the current product; 3) deriving a quantitative baseline; and 4) selecting and evaluating improvement measures.

3. METHOD AND QUALITY MODEL

We developed the GQM-based quality model in two steps: a survey on the relevance of various quality characteristics, and a workshop to identify issues related to the identified quality characteristics, derive GQM measurement goals, and determine concrete metrics for the identified goals. The survey asked the key stakeholders at Ecopetrol to rate the importance of each quality characteristic proposed in ISO 25010 for their projects. Based on their answers, the goal was to focus on the most important quality characteristics during the workshop.

3.1 Survey

The survey included a list of 30 items corresponding to the ISO 25010 quality sub-characteristics, which the responders had to rate using a scale from 1 (very unimportant) to 10 (very important), i.e., a rating of less than 6 indicated minimal importance. The 14 participants were technical leaders from the company's IT department and external software suppliers. To analyze the answers, we simply aggregated the ratings given to the sub-characteristics for the corresponding quality characteristic (Figure 1). Besides the importance of the quality characteristics, we collected data about the type of software developed, the programming languages used, and the areas addressed by the products (Figure 2).

However, we were unable to determine which quality characteristics to focus on because they were all considered very important, with the possible exception of portability. Also, neither a deeper analysis on the level of sub-characteristics nor the grouping of data according to different context factors (such as those listed in Figure 2) revealed any significant difference. Even though the definitions for all quality characteristics and sub-characteristics were provided, it became clear from the feedback of the participants that the different terms were understood in different ways when it came to the technical impact of the quality characteristics on their software. This tends to be a problem with standardized quality models in general; stakeholders understand their particular quality needs but are unable to map them to the general terms provided by the models. So, from the abstract definition of the terms, it was not clear which characteristic is more important than others.

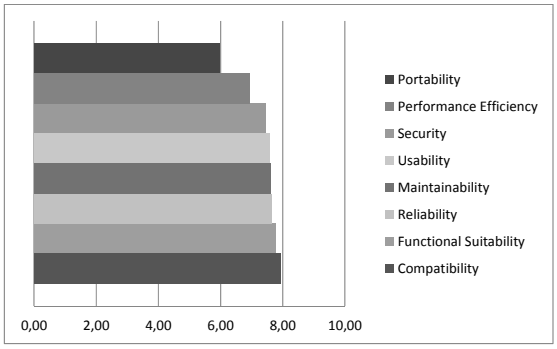

Figure 1. Aggregated rating for quality aspect importance

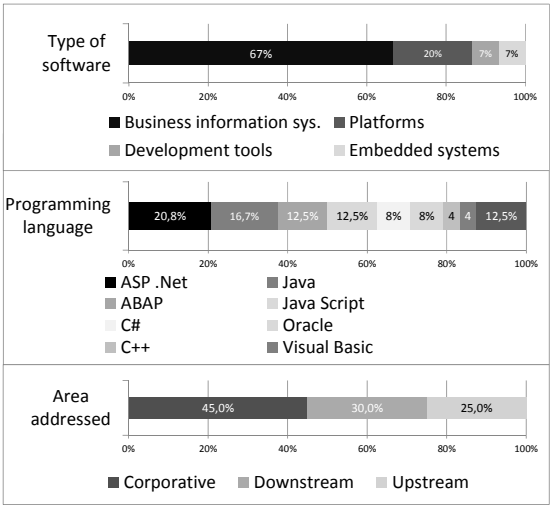

Figure 2. Software type, programing language, and area

For further discussion, we limited the scope of the quality model and characterized the context by asking the following questions:

- What is the application scope of the measurement model?
  - Customization of commercial products and in-house development
  - IS systems only
  - Corporate, upstream, downstream development
- Which organizational units are involved in the measurement?
  - Shared services
- Which stakeholders are involved in the measurement?
  - Architects, technical leads, information leads
- What is the context/environment for the measurement model?
  - Project types: maintenance, new development, integration
  - Development processes: RUP and ASAP (based on SAP)

The next steps, which were carried out during the on-site workshop, were to determine the organizational objectives, prioritize the goals, and develop GQM measurement goals and finally metrics for creating an overall quality model.

3.2 Workshop

Since the survey provided little differentiation among the quality characteristics, it was difficult to prioritize and identify the most

important quality needs. So we chose to take a different approach to gain some more insight into the Ecopetrol "quality" issues of importance. We turned the problem upside down and asked the nine workshop participants to come up with specific issues related to software quality that should be avoided. To this end, we informally applied the UMD approach [3] for eliciting such requirements, which essentially revolves around asking the stakeholders the following questions: What do you see that should not happen, i.e., what causes you the most trouble? What should be done to mitigate this problem? The problem may abstract to several standard quality aspects, e.g., reliability and maintainability, creating some confusion in identifying a particular quality aspect, even though the stakeholder knows what the real quality problem is. So in that sense the defined quality aspect is too abstract to be of value since we cannot map the aspect back to the real problem.

This generated a lively discussion. It was clear that there was a problem with the dynamic nature of requirements elicitation and on-demand implementation of the rules, as business rules and requirements were introduced at random by various levels of stakeholders and multiple implementation teams developing the software solutions. This led to a variety of issues focusing on requirements stability, design and implementation, and code reuse. Some of the issues and the suggested mitigations were:

Requirements Stability:

- **Issue:** Constantly evolving requirements and business rules.
- **Mitigation:** Improve the requirements elicitation process from the customers using more focused reviews and inspections at the right point in time. Try to identify potential future change requests.

Design and Implementation:

- **Issue:** External and internal dependencies of components create problems (if one interface is changed, the whole system does not work or, even worse, the set of information systems interacting with each other).
- **Mitigation:** Make the dependencies (of interfaces) transparent; consider doing an impact analysis.
- **Issue:** Hard-coded parameters in the code.
- **Mitigation:** Identify and eliminate hard-coded parameters; make use of a business rules engine to access the parameters, make better use of the MDM (Master Data Management) system.
- **Issue:** It is difficult to deal with and maintain a very large number of public interfaces between the information systems interacting with each other.
- **Mitigation:** Minimize the number of public interfaces by eliminating deprecated interfaces and unifying the way information systems are interacting with each other.

Code Reuse:

- **Issue:** There can be more reuse; maybe some reuse is achieved, but the process is not transparent.
- **Mitigation:** Create a dictionary of reusable components made visible to all stakeholders.

Once we understood the major issues related to software development, we were able to better identify the quality needs and prioritize where measurement should be applied. From the application of the GQM paradigm [1], we derived ten goals (Table 1).

Other goals were identified based on the issues, but these are not included in the first version of the quality model because measurement data collection for these goals involves a high degree of manual work, e.g., requirements completeness, requirements redundancy, and cost and size of change requests. The focus of the initial model was on goals for which measurement data can be collected at least semi-automatically from existing sources of information. For all ten goals, we applied the GQM approach and derived questions, metrics, and identified variation factors. Table 2 shows an example of a GQM abstraction sheet.

During the discussions in the workshop, many variation factors were identified, including development and support practices, as well as management practices, such as project team fluctuation, regulation changes, operational incidents, project time constraints, number and type of change requests, application performance constraints, application domain knowledge and skills of project team, communication and team work skills of project team, and disciplined requirements management.

Table 1. Overview of GQM measurement goals

Name	Object	Purpose	Quality Focus
G1: Requirements stability	Application requirements	Characterization	Stability
G2: Hard-coded parameters in code (constants, literals)	Application code	Characterization	Hard-coded parameters
G3: Design of internal dependencies	Application design	Characterization	Internal dependencies
G4: Design of external dependencies	Application design	Characterization	External dependencies
G5: Code reusability	Application code	Characterization	Reusability
G6: Code defect density	Application code	Characterization	Defect density
G7: Enterprise architecture coupling	Enterprise architecture	Characterization	Coupling
G8: Test coverage	Traceability matrix	Characterization	Test coverage
G9: Conformance of design to implementation	Design	Characterization	Conformance
G10: Traceability	Traceability matrix	Characterization	Traceability
Viewpoint	**Context**		
Architects, technical leads, information leads	Maintenance, new development, and integration projects, based on RUP or ASAP.		

During the workshop discussion we noted that personnel fluctuation and application domain knowledge and skills have the greatest influence on quality. Since the company's software is developed by external suppliers, managing personnel fluctuation is especially important since it takes time to acquire knowledge and experience regarding the specifics of the oil and gas domain. Personnel fluctuation is one factor for success in a software development project in terms of effectiveness and efficiency. If people with high domain experience/knowledge leave or move to different projects, they will be replaced by people with less domain experience and knowledge and the risk of being less productive typically increases dramatically. In general, it is recommended that personnel fluctuation should be limited as much as possible. As a minimum, a key personnel group should be established that can act as a multiplier of domain experience/knowledge and as advisors for "new" members of the development team.

For each of the ten GQM goals, an abstraction sheet (as shown in Table 2) was developed. After that, all goals, questions, and metrics were integrated into a comprehensive quality model. The initial quality model developed consisted of 10 measurement goals, 17 questions, 36 metrics, and 10 variation factors.

Table 2. GQM abstraction sheet for the design of external dependencies

Object	**Purpose**	**Quality Aspect**	**Viewpoint**	**Context**
Design	Characterize	External dependencies	Architects, technical leads, information leads	Maintenance, new development, integration projects
Quality Focus (Questions and Metrics)				**Variation Factors**
What do you want to know regarding the quality focus? **Q1:** How many of the provided external interfaces are used by other applications per interface type? **M1:** Number of external interfaces provided used by other applications per interface type **Q2:** How many external interfaces are provided per type? **M2:** Number of external interfaces provided per type **Q3:** Which is the proportion of interfaces provided and used by other applications? **M3:** Ratio M1/M2				What explains variations in the quality focus? – Team experience – Team knowledge – Time constraints – Performance constraints
Baseline Hypotheses				**Impact of Variation Factors**
What are known baselines for metrics in the quality focus? (Confidential information)				What is the impact of factors? (Not clearly specified)
Interpretation Model				
How to interpret and assess the data? (Confidential information)				

Moreover, for each goal, the corresponding quality characteristics from ISO25010 were identified and related to the corresponding goals.

4. CONCLUSIONS AND FUTURE WORK

This paper presented our practices and experiences in developing a custom-tailored quality model for Ecopetrol using the classical GQM approach for identifying and measuring information needs. First, a survey was conducted to identify quality characteristics of interest. Second, a workshop was conducted to identify the current top-priority issues and the techniques that can be used to mitigate those problems and improve the quality of the products. Based on this discussion, a set of measureable quality goals that define the oil company's quality focus was defined and agreed upon. For each goal, abstraction sheets were developed to derive concrete metrics that characterize software quality at Ecopetrol. Finally, these metrics were integrated into a comprehensive quality model. Some lessons learned from building the model include:

- Building a quality model is neither a pure top-down nor a pure bottom-up process, and it was easy to use GQM in this context.
- The main drivers for building the quality model were the issues and mitigation strategies identified, rather than the ISO 25010 quality characteristics. The major reasons were that people understood various terms in different ways and the technical impact of the general quality characteristics was hard relate to their specific quality problems. It was easier for people to say what they did not want to happen.
- Using this approach results in a model that may not be compatible with the ones found in other companies (in fact one would assume they would all be different), so it is difficult to do benchmarking across companies. However, the resulting model was understood by and useful to the Ecopetrol stakeholders. Discussion centered on real problems and it was clear when progress was being made in addressing these problems. This reduced the complexity of the quality model and increased acceptance among the Ecopetrol experts participating in the development of the model.

Current work focuses on applying the quality model to a set of selected software development projects at Ecopetrol. Therefore, a tool chain is being created that will automate large parts of the measurement data collection. We are also working on visualizing the analysis results in an intuitive manner, thus allowing for data exploration and identifying root causes for quality issues. In the future, the first analysis of the results will trigger initial recommendations for further improvement of software quality and will lead to the identification of reliable target values for the creation of quality gates as a mechanism for checking software quality as early as possible in the development process. Future work might include further application of the UMD approach for the development of quality models.

5. REFERENCES

[1] Basili, V., Caldiera, G., Rombach, D. 1994. Goal question metric paradigm. In J. Marciniak, editor, Encyclopedia of Software Engineering, Volume 1, pages 528-532. John Wiley & Sons, Inc., New York.

[2] Deissenboeck, F., Wagner, S., Pizka, M., Teuchert, M., and Girard, J.F. 2007. An activity-based quality model for maintainability. In Proceedings of the IEEE International Conference on Software Maintenance, 184-193.

[3] Donzelli, P., Basili M. 2006. A practical framework for eliciting and modeling system dependability requirements: Experience from the NASA high dependability computing project. Journal of Systems and Software, 79, 1 (January 2006), 107-119.

[4] Dromey, G.R. 1995. A model for software product quality. IEEE Transactions on Software Engineering, 21, 2 (Feb. 1995), 146-162.

[5] Dumke, R. 1998. An OO framework for software measurement and evaluation. In proceedings of the Conference on Quality Engineering in Software Technology, 52-61.

[6] ECSS, ECSS-Q-30A. 1996. Space product assurance: dependability.

[7] Grady, R.B. and Caswell, D.L. 1997. Software metrics: establishing a company-wide program: Prentice Hall.

[8] ISO/IEC 25000-1. 2005. Software engineering - Software product quality requirements and evaluation (SQuaRE) - Guide to SQuaRE.

[9] Kitchenham, B.A. 1987. Towards a constructive quality model: part 1: software quality modeling, measurement and prediction. Software Engineering Journal, 2, 4 (July 1987), 105-113.

[10] Kläs, M., Lampasona, C., Münch, J. 2011. Adapting software quality models: Practical challenges, approach, and first empirical results. 37th EUROMICRO Conference on Software Engineering and Advanced Applications (SEAA), 341-348

[11] Rincón, G., Alvarez, M., Perez, M., Hernandez, S. 2005. A discrete-event simulation and continuous software evaluation on a systemic quality model: An oil industry case. Information & Management, 42, 8 (Dec. 2005), 1051-1066.