# Quality Evaluation of Artifacts in Tailored Software Process Lines

Camila Hübner Brondani, Gelson Bertuol, Lisandra Manzoni Fontoura

Programa de Pós-Graduação em Informática
Universidade Federal de Santa Maria – UFSM
Santa Maria, Brazil
chbrondani@inf.ufsm.br; gelson.bertuol@gmail.com; lisandra@inf.ufsm.br

***Abstract* – In software engineering, it is necessary to consider variables such as quality, effort, productivity, time and cost of development. Those variables are negatively affected when defective artifacts are produced. In this case, the cost of rework to correct defects increases in relation to the time of their discovery. Therefore, initiatives should be undertaken in order to find these defects and correct them as soon as they are introduced. This work proposes a mechanism to evaluate the quality goals of software artifacts by means of a quality framework. The study has the objective of organizing concepts that involve the definition of quality goals and their respective methods and metrics of evaluation and can be used to facilitate the task of defining quality plans. Besides that, the framework includes a process to evaluate software artifacts generated from a Software Process Line (SPrL). A Web tool that uses SPrL was used to facilitate the adequacy of the process to different contexts of projects.**

***Keywords-component: Software Quality; Software Artifacts; Process Tailoring; Software Process Lines.***

## I. Introduction

The use of software development processes adjusted to the needs of the project and of the work team has strong influence in the final quality of the produced products. This is due mainly by the fact that the process management and the search for continuous quality improvement tend to generate less defective software and within the expected patterns. On other hand, Al-Kilidar *et al.* [1] affirm that, instead of trying to measure the software quality as a whole, should be sought an evaluation of attributes that compose a product which, when combined, may offer a general notion about their quality.

Among diverse approaches described in literature to define and evaluate the quality of software products, stand out the quality models. They search to structure quality in punctual and easy factors to be analyzed and, at the same time, provide a good characterization of such elements [2]. Quality models can be used to evaluate the final product or the different artifacts produced along with the software development. Quality plans may be created from artifacts selected to compose a tailored process. SPrL turns easy the processes tailoring enhancing quality and adequacy of generated processes, decreasing the risks of an inadequate tailoring and even potentializing the reuse.

Software Process Lines (SPrL) imply that organizational process may be organized according with their similarities and variabilities, allowing the composition of processes based in projects specific needs [3]. In this work, the reuse of elements applied to create a quality plan and the composition of processes are supported in a process line based in the context of the project.

In this sense, this work presents an approach to the evaluation of artifacts generated and/or transformed by several activities that compose a tailored SPrL. The approach has the finality to detect and correct possible problems or defects found previously their spread, reducing rework and improving the quality of final products. The proposal is based in a quality framework, structured from a metamodel that relates the evaluation process to the characteristics that involve the artifacts, such as their purposes, interests, methods and metrics. This work also uses lessons learned to form a repository able to help future evaluations of quality artifacts.

This article is organized as follow: in Section II it is presented important concepts to the understanding of this work. In Section III it is described an overview including: a) a metamodel to evaluate artifacts in tailored software processes, b) an approach to generate tailored software process lines and c) evaluation process of artifacts using a tailored software process line. In Section IV, a case study to illustrate the use of our approach is presented. In Section V presents final remarks and some future steps of this work are also discussed.

## II.Background

Approaches based on knowledge reuse are widely used on software process tailoring, with the purpose of decreasing delivery deadlines and costs, aside from improving the final product quality [4]. The adoption of SPrL allows us to leverage the reuse of individual process components for full process architectures, comprising several inter-related components [3][5][6]. Afterward that, the objective of elaborating consistent processes, once several tailoring approaches are limited to selecting the elements for the process according to the characteristics or the product, and there is no worries with the sequencing and consistency of the generated process. Authors like Jaufman and Munch [7], Washizaki [5] and Barreto [6] propose the use of SPrL as a way to make possible the reuse of software process components.

*The authors would like to thank Fapergs (Fundação de Amparo à Pesquisa do Rio Grande do Sul) for the financial support to this work.*

DOI reference number: 10.18293/SEKE2015-052

At the same time, it is believed that to achieve a final product adequate to the project requirements, the quality should be evaluated along the whole software development process. Quality models are, in general, classified as Hierarchical Models, Conceptual Models and Context Models. The Hierarchical Quality Models describe the relation among a fixed group of high level quality factors, product attributes and appropriate metrics to achieve these factors [8]. They are organized in pillars which are decomposed and refined in specific quality attributes, capable of being evaluated quantitatively by appropriate metrics. The most relevant hierarchical models to this research which, at the same time, where useful as theoretical foundation to the development of others more complex and elaborated models aside from standards and international quality patterns are: the ISO/IEC 9126 [9], the McCall's quality model [10] and Boehm's [11].

The Quality Conceptual Models are used not only in the process evaluation, but as for the development environment as a whole [12]. However, this approach allows not only the evaluation of artifacts, but also the development of team knowledge, the modeled domain, the modeling languages, as for many others aspects that involve the software process building. Two of the most important proposals around the Conceptual Models are the Lindland *et al.* [13] and Krogstie *et al.* [14]. Although the conceptual models may have a high level of abstraction and, therefore, it is more difficult of being practically applied, they can become a good alternative to formalize quality intentions. Other studies have also been proposed to improve the software products and processes qualities during time. These works approaches the evaluation of quality in scientific contexts, normally by means of a framework instantiated from a quality metamodel.

## III. Proposed Approach Overview

This work presents an approach of quality evaluation of software artifacts generated from a tailored SPrL. To achieve this goal, in this section are described: a metamodel with the objective of organizing quality concepts, a systematic process tailoring using SPrL and an evaluation process.

### *A. Metamodel of Evaluation of Artifacts in Tailored Software Processes*

The metamodel (Figure 1) has the objective of helping who is interested in an adoption of a common vision about the requirements of quality intended for a specific project, at the same time that allows a structured decomposition of elements, concepts and relationships necessary to this vision. The metamodel definition was based in three basic requirements, proposed by Trendowicz e Punter [8], which are: flexibility, reusability and transparency.

The created metamodel, called Quality Metamodel for Tailoring Process (QMTP), represents a group of elements considered pertinent for the quality evaluation of software artifacts. The structure, illustrated at Figure 1, as well as its relations, was made from the quality models described in the Section II, and aims on organizing the evaluators' knowledge in the search for evaluation metrics and methods which better represent its quality purposes.

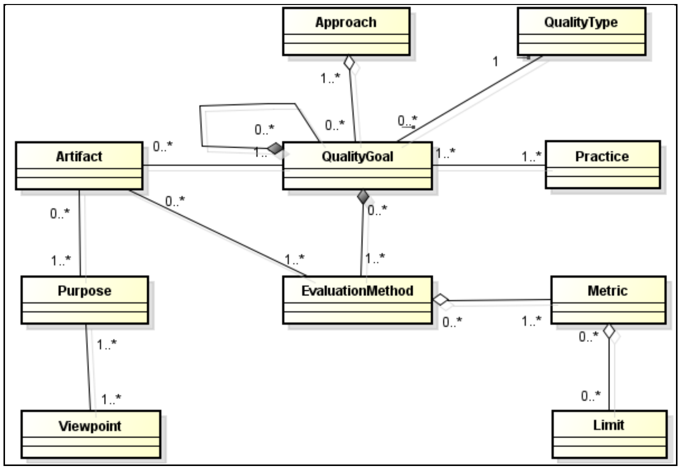


Figure 1. Quality Metamodel for Tailoring Process (QMTP)

The metaclass ***Approach*** is specified to represent instances of different methodologies or paradigms that can be used in the software development. ***QualityGoal*** is the definition clear and comprehensible of which attributes or quality characteristics of a certain stakeholder is interested for a certain software artifact. These goals are specifics for each ***Artifact***. For example, a Use Case Model has as main objective the comprehension of software requirements, while a source code must be complete and consistent. Therefore, in the context of this work, an artifact is basically associated to the delivers which occur during a software process and the propose of formalizing these elements aims, first, at determining which of them can identify the expected quality needs for the product based on the stakeholders expectations.

Thus, to facilitate the evaluation process tailoring and organize the artifacts based on organizational standards, the metamodel allows that the quality goals can be identified by a ***QualityType***. That helps to architecture it on different abstraction levels, which can be created by the evaluation team based on some of the existing models. The ***ViewPoint*** metaclass is used to identify the stakeholders interested in intended quality goals.

At the same time, the ***Purpose*** metaclass has the objective of identifying the purpose that describes the artifacts' intention inside software process life cycle, together with the reason why it should be evaluated. The ***EvaluationMethod*** can be quantitative or qualitative and it identifies how a certain quality goal may be evaluated. These methods are widespread to support specific methods (for example, simulations, inspections, checklists) as well as specific ***Metric***, for example, NUC (number of classes per use case) or NCU (number of use cases per class) for UML models or KLOC (number of errors by a thousand code lines). For this, each metric is defined based on a unit and in the minimum and maximum limit values and a value known as acceptable (***Limit*** metaclass). Each metric has its own particularities. The definition of values and unities, although recommended, it is optional and that the values attributed in the insertion of a certain metric will serve just as reference to the evaluators, which can change it at the moment in which they are defining the quality plan.

### B. *Support Approach in the Software Tailoring Processes*

This project has begun with the work of Lorenz [15], which objective was to define a systematic approach to the software process tailoring from SPrL and information about the projects' characteristics. The approach allows the reuse of process elements, previously defined, enabling the definition of agile and planned processes. In order to validate the work, it was developed a Web tool called Metamodel for Tailoring Process tool (MfTPt) for supporting the tailoring process, improving the element selection technique for reusable processes. This tool has two modules. The main module has all the functionalities for the creation of process elements repository, as: registration of artifacts, tasks, roles, activities, tailoring requirements, attributes for contextualization of project activities and definition of process architectures. The tailoring software process module is systematized by four steps.

1) *Definition of project characteristics:* the criteria for projects contextualization were described from Octopus Model, proposed by Kruchten [16]. In this model, a software project is characterized by the following attributes: size, stable architecture, business model, team distribution, rate of change, age of system, criticality and control.

2) *Selection of the tailoring requirements and of the process architecture:* this step aims to select process elements that will meet those needs and incorporate them into the process. For each component defined in the architecture, activities that have similar purposes are recovered. An architecture from agile or planned approaches can be defined. It is also possible to combine the features of agile and planned methods in a unique process of development to create a hybrid process.

3) *Prioritization of the activities:* the components defined previously for the architecture are retrieved according to the tailoring requirements for creating SPrL. Thus, the components are prioritized by an algorithm based on the technique Analytic Hierarchy Process (AHP).

4) *Creation of the tailored software processes line:* from the defined process architecture and recovered activities, prioritized and selected, there has the development of tailored SPrL according to the situational characteristics of the project. The Figure 2 shows the SPrL for the discipline of requirements.

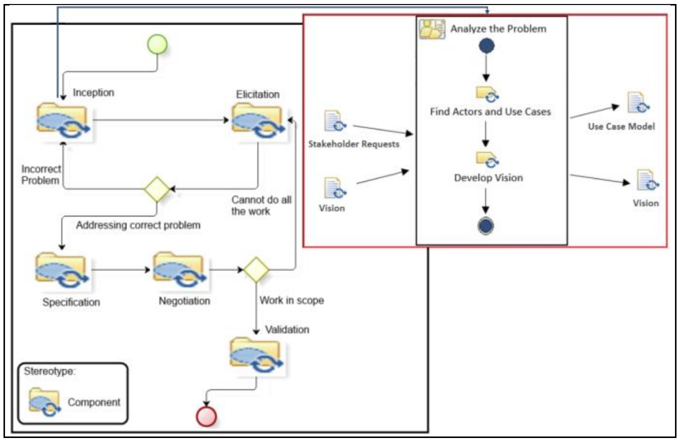


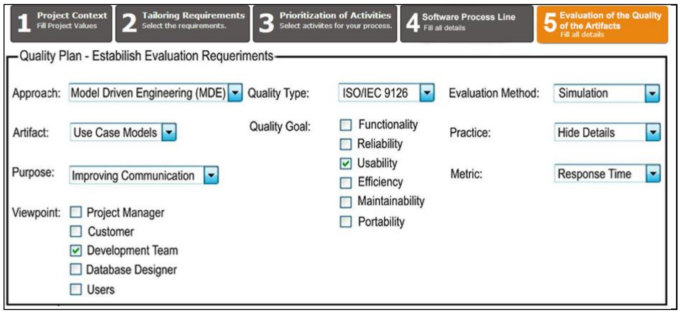
Figure 2. Example of SPrL from requirements discipline

This discipline begins with the component *"Inception"*, goes through the component *"Elicitation"*, *"Specification"*, *"Negotiation"* and *"Validation"*. For each of these components there is one or more selected activities that contain similar situational context. In the Figure 2, it is highlighted as an example of component the activity *"Analyze the Problem"*. This activity is selected to be instantiated in the component *"Inception"*, with its input and output artifacts.

### C. *Tailored Software Process Line Artifacts Evaluation Approach*

This section describes, as part of the proposed quality framework, an evaluation process for software artifacts of the tailored line. The process has been built upon the standard ISO/IEC 14598 [17]. The objective was to organize the data, structured by the metamodel, so that the evaluators have a solid and practical reference at the time to validate the artifacts produced during the project based on quality goals they think are the most important to the final product.

As a first step to determine the quality plan, there is a need to populate the repository. The data can be extracted from many different sources, including models, specific work or even experts experience or previous projects developed by the organization. You should then define the elements of quality, and for each item there are instances already registered and the possibility of keeping each one of them or inserting new ones. In the Figure 3 there is an example on the use of the module Evaluation of the Quality of the Artifacts of MfTPt according to Figure 1.



Quality Plan - Estabilish Evaluation Requeriments

Approach: Model Driven Engineering (MDE)

Artifact: Use Case Models

Purpose: Improving Communication

Viewpoint:
☐ Project Manager
☐ Customer
☑ Development Team
☐ Database Designer
☐ Users

Quality Type: ISO/IEC 9126

Quality Goal:
☐ Functionality
☐ Reliability
☑ Usability
☐ Efficiency
☐ Maintainability
☐ Portability

Evaluation Method: Simulation

Practice: Hide Details

Metric: Response Time

Figure 3. Example of the use of the module Evaluation of the Quality of the Artifacts of MfTPt

It was proposed an instantiation of objects for the evaluation of models of Use Cases in order to improve communication of a software project developed under the model-driven paradigm. The goal of main quality is usability, based on internal quality proposed by ISO/IEC 9126 [9]. However, the choices of what purposes, evaluation methods, metrics and practices are best suited to each situation is the prerogative of quality analysts.

The step that corresponds to the evaluation process begins with the definition of the life cycles that make up the used process model. From each phase of the project the assessment requirements are established for the definition of the quality evaluation plan. That is, for each phase of the process life cycle, the generated artifacts are selected, and for each artifact, the evaluators define its purpose, stakeholders and related quality goals. Then there is the review of the specification that involves relate evaluation methods, metrics and practices for each selected artifact.

The last step comprises the documentation of procedures defined and that will be used by the evaluators to define the quality of the selected artifacts, that is, at this stage it produced the quality plan. The idea is that all artifacts settings and their relationships are structured in a clear and understandable way in order to guide the evaluators in quality validation. Finally, the quality plan will be stored and can serve as reference for future assessments, as the artifacts generated during a software process are similar to keeping the same development approaches and the same models of software process.

## IV. Case Study

In order to validate the proposal put forward in this paper, we performed a case study involving the evaluation of quality in a software project, developed by undergraduate students of Information Systems at the Universidade Federal de Santa Maria (UFSM). Though it had academic purposes, the project was implemented and deployed in a client. An incremental development process was adopted, consisting of development cycles, where each cycle in the following activities were carried out: definition and requirements analysis, design, coding and testing. The following artifacts were generated: Project Charter; UML models – Use Cases, classes, components diagrams and data model, in addition to source codes.

The project aims to develop a software to manage the payment of mentoring grants of an education institution called Instituto Federal Farroupilha (IFF). This institution has eight geographically distributed campuses. The application's goal is to computerize the management system and the payment of fellow teachers who work on campus.

The case study took place in two stages. The first involved the undergraduates who developed the project and the second had the support of students of graduate in Computer Science from the same university. Training was offered addressing concepts of quality, existing quality models, importance of assessing the quality of software products (and their artifacts as well) and the objective of this work was explained. Then, in possession of a handout provided, the project team aimed to point out, in the group opinion, what were the most important quality goals for the project as a whole and for each of the artifacts created during the project, and stakeholders roles and what means known to evaluate the defined goals.

In response, there was some compatibility between the quality goals chosen in the first phase with those presented in the second phase of the experiment. However, respondents indicated having a high degree of difficulty to suggest possible methods of assessment for each quality goal. The groups were asked regarding the importance of evaluating the quality of software products and as how much a specialized tool could help development teams in the definition and implementation of quality in these products. In response, respondents converged their ideas arguing that the assessment of quality can lead to reduced incidence of errors, reducing costs and production time and positively impact the developer organization's image. At the same time, there was consensus that an aid tool in defining and assessing the quality of software products is essential in the application of quality concepts.

## V. Concluding Remarks

This work presented a proposal of artifacts evaluation process that is inserted in a SPrL tailored from the reuse of properly characterized activities. The SPrL tailored uses a process architecture for which activities to compose the SPrL are retrieved and prioritized. The Web MfTPt tool was developed to support the tailoring process, improving the technique of selecting the reusable process elements and helping the process sequencing, optimizing the resources and improving the process management. The quality plan for artifacts evaluation is developed considering the group of artifacts selected during the process tailoring process and from the reuse of instances of the quality metamodel.

The validation of the proposed approach was accomplished by means of a case study involving a software project. Between the future works, it is intended to validate the proposed approach in real projects, in addition to develop an automation of elements that comprise the artifacts evaluation plan, because it is dependent on human intervention, in other words, on the evaluators knowledge and experience.

## References

[1] H. Al-Kilidar, K. Cox, and B. Kitchenham, “The use and usefulness of the ISO/IEC 9126 quality standard,” *International Symposium on Empirical Software Engineering*, pp. 126–132, 2005.

[2] F. Deissenboeck, E. Juergens, K. Lochmann, and S. Wagner, “Software quality models: purposes, usage scenarios and requirements,” *7th International Workshop on Software Quality*, pp. 9–14, 2009.

[3] D. Rombach, “Integrated software process and product lines,” *International Conference on Unifying the Software Process Spectrum*, pp. 83–90, 2005.

[4] L. M. Northrop, “SEI’s software product line tenets,” *IEEE Software*, vol. 19, no. 4, pp. 32–40, 2002.

[5] H. Washizaki, “Building software process line architectures from bottom up,” *Product-Focused Software Process Improvement*, vol. 4034, pp. 415–421, 2006.

[6] A. S. Barreto, L. G. P. Murta, and A. R. C. da Rocha, “Software process definition: a reuse-based approach,” *Journal of Universal Computer Science*, vol. 17, no. 13, pp. 1765–1799, 2011.

[7] O. Jaufman and J. Münch, “Acquisition of a project-specific process,” *Product Focused Software Process Improvement*, vol. 3547, pp. 328–342, 2005.

[8] A. Trendowicz and T. Punter, “Quality modelling for software product lines,” *7th Workshop on Quantitative Approach in Object-Oriented Software Engineering*, 2003.

[9] ISO/IEC 9126, “Software Engineering - Product Quality,” 2003.

[10] J. A. McCall, P. K. Richards, and G. F. Walters, “Factors in Software Quality,” *Nat’l Tech. Information Servicel*, vol. 1, 2 and 3, 1977.

[11] B. Boehm, J. R. Brown, H. Kaspar, M. Lipow, G. McLeod, and M. Merritt, “Characteristics of software quality,” *TRW of software technology*, 1978.

[12] K. Mehmood, S. S. Cherfi, and I. Comyn-Wattiau, “Data quality through conceptual model quality-reconciling researchers and practitioners through a customizable quality model,” *14th Internation Conference on Information Quality*, pp. 61–74, 2009.

[13] O. I. Lindland, G. Sindre, and A. Solvberg, “Understanding quality in conceptual modeling,” *Software, IEEE*, vol. 11, no. 2, pp. 42–49, 1994.

[14] J. Krogstie, O. I. Lindland, and G. Sindre, “Defining quality aspects for conceptual models,” *International Working Conference on Information System Concepts: Towards a Consolidation of Views*, pp. 216–231, 1995.

[15] W. G. Lorenz, M. B. Brasil, L. M. Fontoura, and G. V. Pereira, “Activity-based software process lines tailoring,” *International Journal of Software Engineering and Knowledge Engineering*, vol. 24, no. 9, pp. 1357–1381, 2014.

[16] P. Kruchten, “Contextualizing agile software development,” *Journal of Software: Evolution and Process*, vol. 25, no. 4, pp. 351–361, 2010.

[17] ISO/IEC 14598, “Information Technology - Software Product Evaluation,” 1999.