Artifact Quality Assessment Plans Generation from Tailored Processes

Camila Hübner Brondani, Gelson Bertuol, Lisandra Manzoni Fontoura
Departamento de Computação Aplicada – DCOM
Universidade Federal de Santa Maria – UFSM
Santa Maria, Brazil
{chbrondani, gelson.bertuol, lisandramf}@gmail.com

***Abstract*— The production of quality software is a basic and essential Software Engineering goal. A software product quality assessment should be started at the early stages of a development process, to detect and correct problems before they propagate, making their correction more expensive. For that to be possible, it is necessary to assess the quality of each artifact generated during the development process, to allow the production of defect-free artifacts improving the final product quality. In this work, we propose an approach for the generation of quality plans during the tailoring of software process. When the user selects the quality practices to be used in a project, a set of activities satisfying those practices are inserted in the project's software process, along with their associated artifacts. Our goal is to define the quality assessment plans for these artifacts. The approach was validated through case studies of real projects in different companies, involving experts with large software development experience. The interviewees analyzed the approach and considered the proposal of this work as positive because it facilitates the definition of assessment plans, the plans are adequate to the selected criteria and that quality control during the process decreases rework.**

Keywords*— *Software Quality; Software Artifacts; Process Tailoring; Quality Practices.

I. Introduction

It is consensual that software quality is highly influenced by the adoption of a software development process, this is mainly due to the management practices and the continuous pursuit for higher quality brought in by the processes, leading to software with fewer defects. The quality assurance activities aim to evaluate artifacts quality at each development stage, avoiding error propagation. Al-Kilidar et al. [1] state that, instead of trying to measure the quality of a software system as a whole, one should try to evaluate each intermediate software development product which, when combined, may provide for a broad idea about the whole system quality. However, even though quality is a recurrent issue in Software Engineering, most organization lack experts capable of defining their product's desired quality features. Furthermore, quality features definition and classification alone are not sufficient without a discussion about the means to reach those features and the relevant roles involved [2].

In this paper, it proposes a systematic approach for defining a quality assessment framework for artifacts generated or transformed by the activities that make up an tailored software process. The framework is composed by a metamodel, a knowledge base based upon the CMMI quality model [3], an assessment process and a supporting tool.

Briefly, the process engineer selects a set of quality practices that should be satisfied by the project. Practices are associated with activities that are recovered from the knowledge base and inserted in the project-specific process. For each activity, a set of artifacts is described, which are then evaluated regarding their quality elements using the ISO/IEC 9126 [4]. These, in turn, are described in a metamodel that links the assessment process to artifacts characteristics such as their purposes, their stakeholders, their corresponding methods and metrics. The final result is the definition of quality assessment plans for the artifacts described in the tailored software process.

These plans are based on the idea that a software product quality assessment may be started at the first stages of a software process [5]. Besides, they help the process engineers who need to customize the process for specific projects, allowing an organization to implement a set of quality practices from a stored knowledges base.

This paper is divided as follows: Section 2 presents the proposed framework, which includes: a) a metamodel, b) a knowledge base, c) an artifacts quality plan implementation process and d) a support tool. Section 3 shows the case studies. Finally, Section 4 presents our thoughts, conclusions and future work perspective.

II. The Definition of an Artifact Quality Assessment Plan

In this work, quality plans to assist evaluators, engineers and other stakeholders in the validation of software artifacts, are generated as a result of a process tailoring. For this, a set of tasks was proposed, as shown in Figure 1.

Initially, it is necessary to inform the characteristics of the process (task 1) and the process architecture that will be used in the adaptation (task 2). The next step is to select quality practices (task 3). Based on this information, a set of activities that meet the reported criteria will be retrieved from the knowledge base (task 4). The process engineer should select the activities that he wishes to include in the tailored process (task 5). When selecting the activities, the tailored process is created (task 6). In the next task, quality assessment plans for the artifacts are generated (task 7).

The tailoring is based on a process framework that integrates the necessary elements to instantiate tailored processes and to define quality plans. This framework is composed by a metamodel that presents the set of elements considered relevant for software artifacts quality assessment (described in section A), a knowledge base (described in section B), an assessment process (described in section C) and a supporting tool (described in section D).

DOI reference number: 10.18293/SEKE2019-032

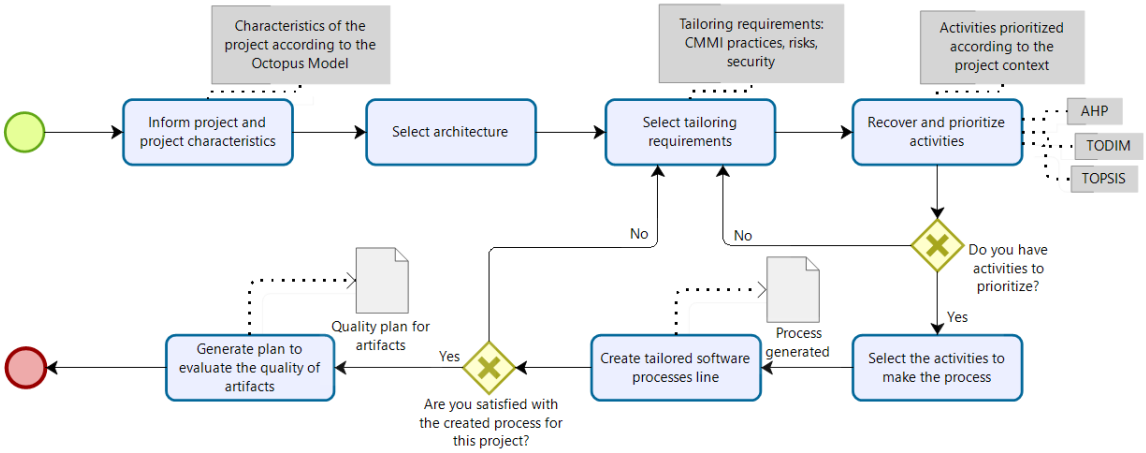

Fig. 1. Modeling Software Process Tailoring Approach to Quality Assessment of Artifacts

A. *The Metamodel*

The Quality Assessment of Artifacts in Process metamodel (QAPro-M) aims to provide and structure both the processes elements and the quality elements to be used in the assessments, helping the stakeholders to achieve a common vision of the quality requirements of a given project. It also allows for a structured decomposition of the elements, concepts and relationships necessary for this vision. The metamodel definition was based upon three basic requirements proposed by Trendowicz et al. [6]: flexibility, reusability and transparency.

The flexibility is associated to the software quality dependence on the context. The assessment model must be flexible enough to adapt to the different approaches resulting from each software project's organizational environment. For the present work, flexibility also refers to the differences among the very artifacts produced during a software development project phases. The documents, UML models, source code and other artifacts, each have their own particular characteristics. The assessment framework must then allow the evaluators to identify those characteristics and define which assessment proposals are best for each of them.

At the same time, reusability is associated to the preservation of knowledge from past experiences and its use on future projects, with a direct impact on development time and cost and, consequently, on the projects profitability. Naturally, reusability in an assessment project depends on projects similarity. Nevertheless, reuse may also refer to the measured data and to quality features and their relationships. Reuse also allows for model refinement, making it more precise and efficient.

Finally, an assessment model must provide a rational and transparent analysis about the relationships between quality features and sub-features, and about how they affect each other. For instance, the development team must be able to see how a class diagram modularization – that will later be used to define the database tables – allows for a better understanding of the software structure. However, it is a known fact that over-normalization may impair database performance, affecting the whole system. One solution to circumvent the transparency issues is to allow the interested parties themselves to define, consensually, the most relevant assessment metrics and methods, the ones that better represent each artifact, using the metamodel to reduce or eliminate any ambiguities and redundancies.

The QAPro-M, depicted in Figure 2, is composed by 23 classes. The metamodel central class is the **activity**. A set of activities from the same area is a **discipline**. Disciplines are distributed along interactive **phases** that sum up to form the development process **lifecycle**. An activity is composed by tasks, each **task** containing one or mode roles. A **role** is a function or job carried out by a person in a project. The **project** class represents the projects defined for an **organization**. An organization may then have many projects, and each project may have many processes. This way, if a **process** does not fulfill the organization expectations, it may be evolved, the process new version being created based on the current one. When creating a new project or activity, it is possible to define a **situational context** for them. For project contextualization we used the Octopus Model that describes the following contextual factors: size, stable architecture, business model, team distribution, rate of change, system age, criticality and governance [7].

The process tailoring takes the activities and the tailoring criterions into account. Each activity has one or more **tailoring criteria**, whose function is to define which requirements may be satisfied by that activity instantiation, retrieving from the database the activities satisfying those criteria.

As a quality model, we chose to use CMMI, each organization can use the most appropriate model for their needs. Then, each **maturity level** is composed by a set of **process areas**. These, in turn, are composed by **specific goals**. Each specific goal is composed by a set of **specific practices**.

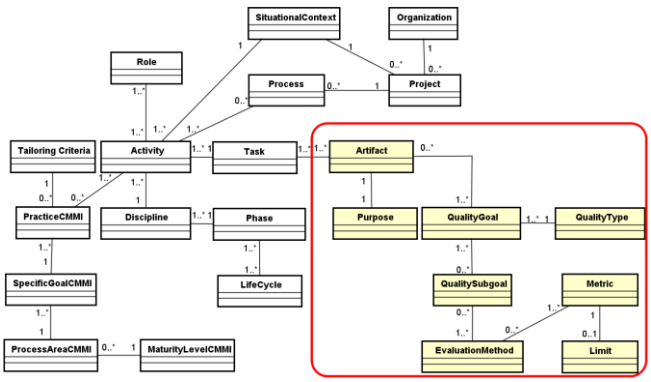

Fig. 2. Quality Assessment of Artifacts in Process - Metamodel (QAPro-M)

The **artifact** class represents the tasks outcomes. Artifacts are linked to a **purpose**, which identifies the artifact's intention and purpose inside the lifecycle, as well as the reasons why that artifact should be assessed.

The quality plan, on its turn, comprises the set of elements responsible for the artifact's assessment. The plans are related to the **quality goal** class. A quality goal represents an artifact's quality features or attributes of interest from a specific stakeholder. The metamodel allows for the quality goals to be identified by a **quality type**. ISO/IEC 9126 is an instance of a quality type, having 6 features, each feature subdivided into **quality subgoals**. The **evaluation method** identifies how a certain quality subgoal should be evaluated. These methods are generalized to allow for a specific methods and **metric**. Also, each metric defines a **limit** value as its acceptable value.

B. Knowledge Base

The first step towards defining the quality plan was to populate the knowledge base. Data was gathered from scientific literature, and included specific models and researches. The knowledge base is expandable, and may be grown based on the expert's experience from past projects.

In order to include quality-focused tailoring requirements, the CMMI quality model practices were incorporated to our approach. This way, the processes engineer may choose the desired organizational maturity level to be attained, the process area, the specific goal and, finally, the CMMI practices related to the goal to be reached by the process. Then the activities needed to satisfy the chosen practices are retrieved from the knowledge base in order to build the tailored process.

We started by analyzing the Requirements Management and Configuration Management process areas. We chose Requirements Management, since a project without well-defined and managed requirements has a far less probability of reaching its goals. Therefore, ensure the management of requirements is paramount to a project success. Likewise, Configuration Management is essential in order to produce quality software, since changes during development are inevitable. The practices described by CMMI were connected to activities and artifacts capable of satisfying each practice.

In order to organize the activities stored in the knowledge base in software processes, we used process architectures. Architectures are composed by the elements used to define a software process. They define the "skeleton" that a process must have, establishing the main elements and how they relate to each other [8]. In Figure 3, the architecture defined for the Requirements and Configuration and Change Management discipline can be viewed.

For each component, one or more activities are retrieved. The activities are prioritized using multi-criteria techniques such as AHP (Analytic Hierarchy Process), TODIM (an acronym in Portuguese for the Brazilian developed Iterative and Multi-criteria Decision Making Method) and TOPSIS (Technique for Order of Preference by Similarity to Ideal Solution), taking into account the activity and the project's situational context.

Afterwards, a specific process for the project is composed by retrieving elements previously included in the process architectures. So, using architecture allows for the retrieval of a set of activities prioritized accordingly to the project context and tailoring requirements. From the process architecture previously defined and the activities retrieved, prioritized and selected using the mathematical methods, it is possible to create the tailored process.

In previous work [9] we describe the process tailoring approach in detail. In this work, we extend this approach to incorporate process tailoring using quality criteria and to generate quality plans for assessing software artifacts.

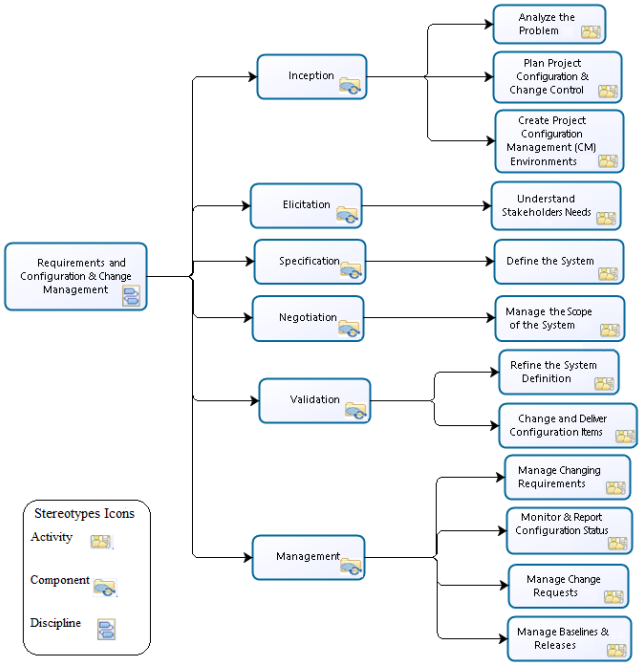

Fig. 3. Architecture for the Requirements and Configuration & Change Management discipline

C. Quality Plans Elaboration Process

The artifacts quality assessment process was build based on the ISO/IEC 14598 [10]. This standard was chosen both for its compatibility with the concepts laid out by the quality models and because it describes an assessment process for the quality features described by the ISO/IEC 9126 [4]. Some ISO/IEC 14598 sub-processes were adapted to reflect specific aspects of the software artifacts assessment.

The assessment process proceeds through the following steps: a) define the assessment requirements (defining what should be assessed); b) specify the assessment (selecting the goals or quality features related to the assessed object detailing assessment methods, metrics, limits and practices for each artifact); c) design the assessment (producing the assessment plan including the documentation of the procedures previously defined that will be used later to define the selected artifacts quality).

ISO/IEC 14598 last phase is the assessment execution, consisting of the products and its components inspection, measuring and testing, according to the assessment plan. As this is an execution-dependent task, not related to the process definition, it falls beyond the scope of this paper.

At the end of phase 3, the data describing the quality plans defined during the specific project assessment process

may be stored for use as a reference for future projects assessments.

As an example of the quality plan creation and the proposed approach workings, let us consider the user selected the CMMI practice “Obtain Commitment for the Requirements”. This practice belongs to the “Manage Requirements” Specific Goal, which in turn is part of the “Requirements Management” Process Area described in CMMI maturity level 2. The Typical Working Product for this practice is the “Documented commitments to requirements and requirements changes”.

Once this CMMI practice is selected, the knowledge base is searched for the activities that satisfy the practice. For instance, the “Manage Change Requests” and the “Manage Requirements Changes” activities from the RUP [11] disciplines “Configuration and Change Management” and “Requirements” will be retrieved and prioritized. After the activity’s prioritization, if this activity is selected to compose a tailored process, the goal would then be the generation of execution plans for the artifacts and process activities quality assessment.

Each activity has one or more input and output artifacts linked to a quality plan, describing how to assess them considering specially its features, objectives, stakeholders, assessment methods and metrics, in order to improve the final product’s quality. Figure 4 shows the metamodel instantiation for the “Change Request” artifact from the “Manage Change Requests” RUP activity assessment. This artifact’s objective is to document and follow the product’s change requests. The main quality goal is maintainability, based on the ISO/IEC 9126-proposed internal quality. From there the analyzability and changeability sub-goals may be explored. These goals and sub-goals may be assessed by specific methods and quantified by related metrics.

The quality plans main task if to organize the data structured by the metamodel, so the evaluators have a solid, clear and understandable reference when assessing the project’s artifacts, based on the quality goals they think are the most relevant for the final product.

D. QAPro System Support Tool

The Quality Assessment of Artifacts in Process (QAPro System) tool was developed from as extension of the MfTPt tool [9]. For the quality plans to be generated, instances of quality elements (described in QAPro-M) must have been defined and stored in the knowledge base.

The first activity of the systematic is the contextualization, for this we used the Octopus Model, allowing for the definition of values for each of its eight factors in the project (Figure 5-A). After that, it is necessary to define the desired architecture for the process tailoring.

The selection of requirements to be considered in the tailoring process are the Specific Practices of the Requirements Management e Configuration Management CMMI process area, shown in Figure 5-B.

Figure 5-C shows the architecture activities retrieval and prioritization according to the project context and the tailoring requirements, using the AHP, TODIM and TOPSIS methods. The process engineer selects the activities.

Finally, the last task consists in the generation of the plan to assess the created process artifacts. This step proceeds as follows: for each activity in the created process, a task is selected and then the artifact for which the quality assessment plan is to be obtained. The assessment specification for each selected artifact is then shown, with its objective, its quality goals and sub-goals, the quality type, the assessment method as well as the metric and the eventual limit value used for this artifact assessment (Figure 5-D).

This documentation will be used later by the quality analyst to define the selected artifacts quality. The idea behind the plan is to present quality elements in a clear and understandable way, to allow it to serve as a guide for evaluators during the process artifacts quality assessment.

However, the evaluators are free to modify any quality element according to the artifacts quality assessment needs. Thus, the flexibility and transparency requirements are kept during the assessment process.

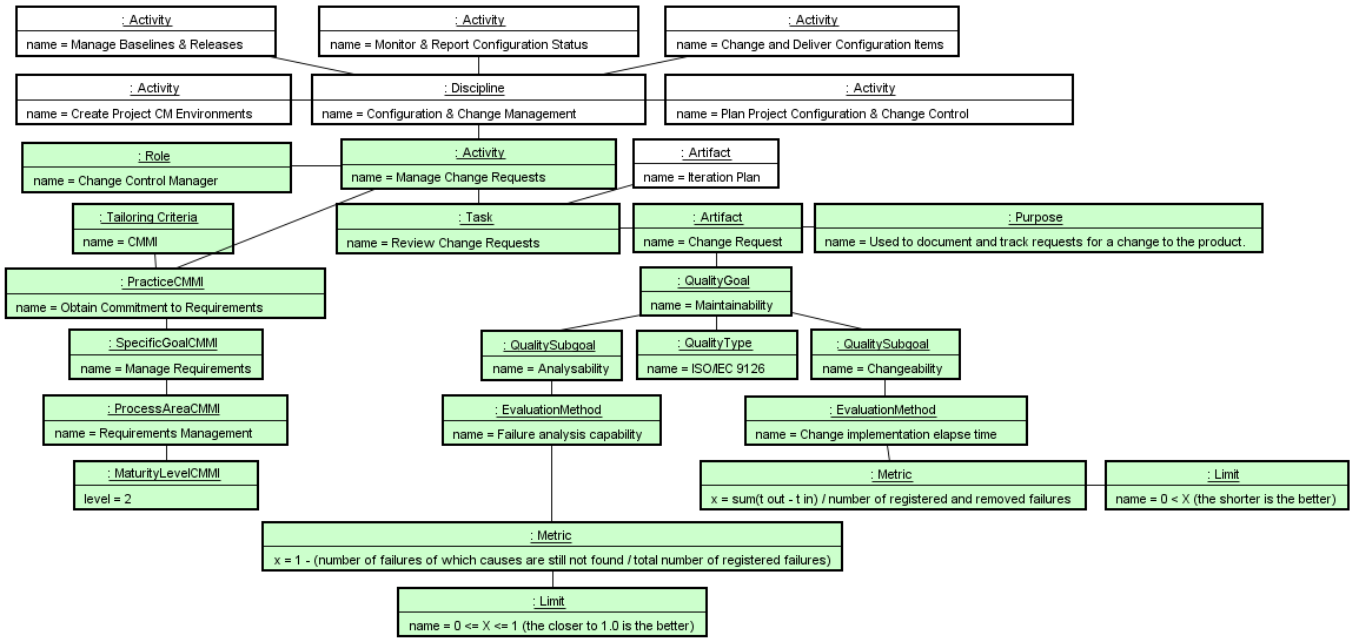

Fig. 4. Instance of the QAPro-M

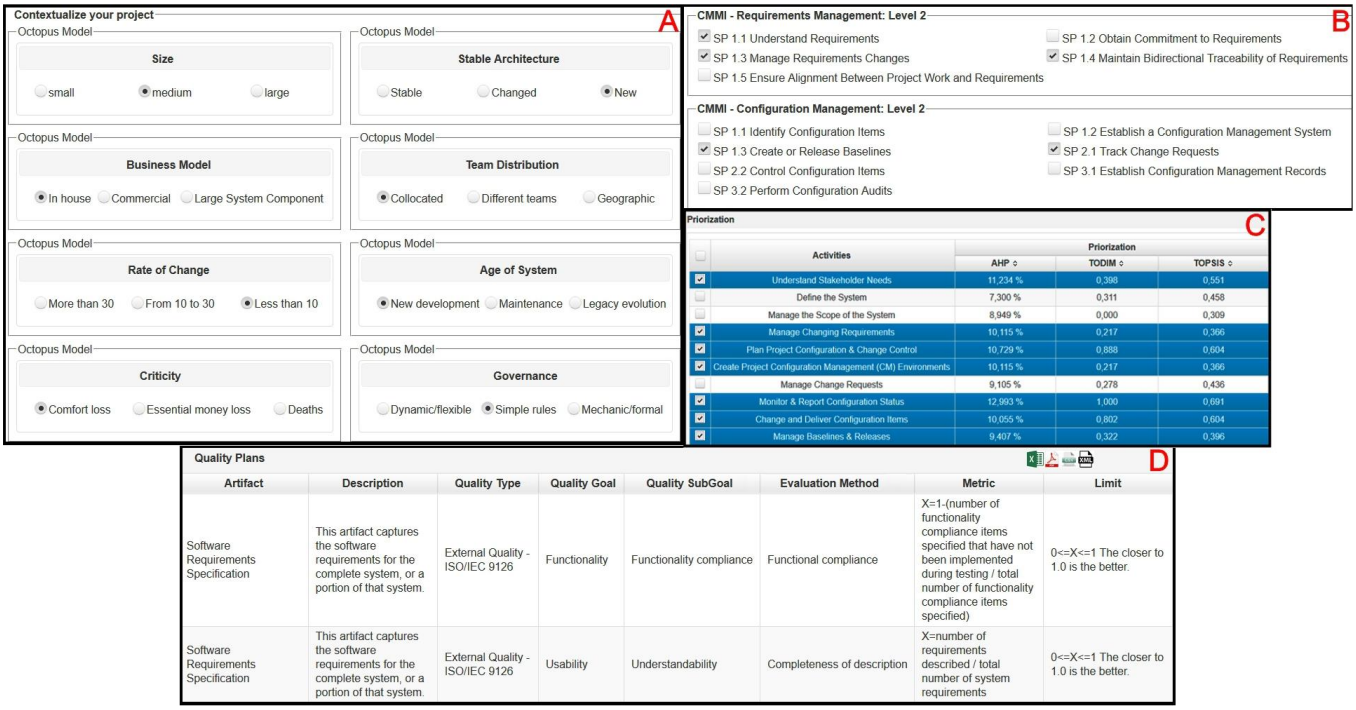

Fig. 5. Quality Assessment of Artifacts in Process (QAPro System)

III. Case Study

In order to validate this proposal, we carried out five case studies involving different real software projects from different companies. The case studies were carried out by three project managers, a compliance analyst and a systems analyst, so we could obtain a point of view from professionals in the field.

A. Case Study Configuration

The case studies were carried out in projects with different business domain. Each case study was divided in three phases:

Data collection: the participant (project member) filled a form identifying your profile, the project context and the tailoring requirements to be satisfied by the process.

Tailored process creation and quality plans generation: with the data informed and using the QAPro System support tool, the process was defined to meet the project's requirements and quality plans were generated for each artifact selected for the process. The results were sent to the participant for evaluation.

Interview and analysis: the participant experts were interviewed to evaluate the generated process and plans. The interview had 15 questions divided in 3 topics: software process, software quality and artifacts quality plans. Each interview lasted approximately 40 minutes, depending on the expert. After that, the results of the case studies were analyzed aiming to evaluate the applicability of the work.

B. Case Study – ASTROS Integrated Simulation System

To illustrate the validation process, this case study is described in detail. This project aims at the research and development of a virtual tactical simulator for military training. The interviewee was the project manager.

The project was characterized by the following attributes: a) Size (Medium); b) Team Distribution (Collocated); c) Criticality (Comfort Loss); d) Stable Architecture (New); e) Rate of Change (Less than 10); f) Governance (Simple rules); g) Business Model (In house) e h) Age of System (New development). The specific CMMI practices chosen were: Understand Requirements, Obtain Commitment to Requirements, Manage Requirements Changes, Identify Configuration Items, Establish a Configuration Management System, Create or Release Baselines, Establish Configuration Management Records, Perform Configuration Audits.

After the prioritization methods results analysis, the following activities were selected to compose the process: Plan project configuration & change control, Understand stakeholder needs, Define the system, Change and deliver configuration items, Monitor & report configuration status, Manage change requests, Manage baselines & releases.

Quality plans were suggested for the following artifacts: Configuration Management Plan, Stakeholder Request, Software Requirements Specification, Supplementary Specification, Workspace, Configuration Audit Findings, Change Request, Test Results, Test Log and Work Order.

These quality plans along with the tailored SPrL were analyzed by the participant before the interview. The interviews results are discussed jointly in the next section.

C. Case Studies Discussion

The interview first part concerned the software development processes, centered around the following topics: presence of a well-defined process in the company, process tailoring use, agile or planned process use and software artifacts.

It was found that only one of the five projects has a development process with well-defined activities and artifacts, with a planning phase where the process compliance team analyses the project to verify its adherence

(or its lack of adherence) to agile practices. This is carried out through the use of checklist that evaluates, among other things, the project size, projected duration, team size, definitions and architecture patterns.

The goal of the interview second part, about software quality, was to discover if the organization uses any quality model, if a defective artifact can delay the project or raise its cost and if, by selecting CMMI practices as proposed here, there were improvements in the software process. The answers about quality models were unanimous, no project use them. All the participants stressed the difficult to adequate the model to the project's reality. As for defective artifacts, some participants mentioned cases from their own organizations showing that they may indeed delay and make projects more expensive.

As for the CMMI practices selection by the experts, all found it beneficial, as the foreknowledge of the practices to be followed allows for a better planning of the artifacts that would allow these practices to be attained. They found it a simple way to use the CMMI, with an intuitive approach that facilitates process creation. Also, the automated activities generation also saves effort and time, by replacing the need for a deeper model analysis.

The third topic goal was to understand the importance of the software artifacts quality assessment, which teams found the quality plans more efficient and if the artifact quality assessment plans format was satisfactory. This topic results analysis show that the proposed plans may be used by both large and small teams, with the adequate metrics varying according to the team and the project. One participant felt that the plans may be more important for large and distributed development teams, as the importance of documentation tends to grow, as well as the need for better artifacts.

All experts said that the proposed quality assessment plans allow for an artifact's assessment. They found the plans well-organized and easy to understand, and appreciated the plans clear separation of CMMI data, RUP data and artifact assessment.

IV. Conclusions and Future Work

The benefits of developing quality software products are diverse, widespread and well known. However, the importance of evaluating the quality of these products goes beyond commercial or security issues, because it aims to provide qualitative and quantitative results on the quality of the software produced. In addition, provide the necessary feedback for the improvement of the software processes. But to be effective, these results must be understandable, trustworthy, and in keeping with the environment surrounding evaluations.

This work showed a framework for quality assessment of artifacts created or changed by activities forming a tailoring software process. For each process component, different activities may be selected, sharing a similar situational context and covering the desired quality practices to be used by the tailored process. These activities have artifacts, which in turn are associated to the quality plans defined here. The artifact assessment plan is elaborated taking into account the set of artifacts selected for the tailored process and reusing instances of the quality metamodel. It should be noted that the generated quality plan serves as reference for the process engineers, who can, however, modify the plan if necessary. Besides that, the quality plans should evolve through time, along with the organization's maturity.

The approach validation was carried out using case studies. The participants agreed that the proposed approach to assess the software development generated artifacts is valid and relevant.

Future work includes the use of quality plans in real projects by monitoring quality throughout the project. To address one of the limitations observed, namely the use of activities found in planned processes, we also suggest the association of the quality practices to other groups of activities, such as the ones found in agile practices and methods. So it would be possible to create quality assessment plans with agile methods metrics, such as progress and productivity.

Acknowledgment

We thank the Fapergs (Fundação de Amparo à Pesquisa do Rio Grande do Sul) and the Brazilian Army for the financial support through the SIS-ASTROS Project (813782/2014), developed in the context of the PEE-ASTROS 2020.

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] P. Mohagheghi, V. Dehlen, and T. Neple, "Towards a Tool-Supported Quality Model for Model-Driven Engineering," *Proceedings of the 3rd International Workshop on Quality in Modeling*, 2008.

[3] SEI, "CMMI® for Development, Version 1.3," 2010.

[4] ISO/IEC 9126, "Software Engineering - Product Quality," 2003.

[5] T. L. Dubielewicz L, Hnatkowska B., Huzar Z., "Software Quality Metamodel for Requirement, Evaluation and Assessment," *ISIM'06 Conference*, pp. 115–122, 2006.

[6] A. Trendowicz and T. Punter, "Quality Modeling for Software Product Lines," *7th Workshop on Quantitative Approach in Object-Oriented Software Engineering*, 2003.

[7] P. Kruchten, "Contextualizing Agile Software Development," *Journal of Software: Evolution and Process*, vol. 25, no. 4, pp. 351–361, 2013.

[8] 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.

[9] 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.

[10] ISO/IEC 14598, "Information Technology - Software Product Evaluation," 1999.

[11] IBM Rational, "Rational Unified Process: Version 7.2," 2003.