Representation of Quality Attribute Techniques Using SPEM and EPF Composer

Yin Kia Chiam¹,²,³, Mark Staples¹,², Liming Zhu¹,²
¹NICTA, Locked Bag 9013, Alexandria NSW 1435, Australia
{yinkia.chiam, liming.zhu, mark.staples}@nicta.com.au
²School of Computer Science and Engineering, K17, University of New South Wales, Sydney, NSW 2052, Australia
³School of Computer Science, University of Malaya, 50603 Kuala Lumpur, Malaysia

Abstract

There are many development techniques used to assist development teams to achieve required levels of product quality such as safety, security and performance. These "Quality Attribute Techniques" (QAT) aim to identify, eliminate, reduce, control and minimise potential quality problems in the development of critical systems. Although widely used, these techniques are not normally well represented in software process models. This paper proposes two alternative representations of Quality Attribute Techniques using the SPEM metamodel and Eclipse Process Framework (EPF) Composer and shows how these techniques can be incorporated into software development process models. Safety techniques have been selected as a case example for evaluation. The evaluation identifies advantages and limitations of the SPEM and EPF Composer in terms of their ability to support representation and integration of Quality Attribute Techniques. Some improvements to SPEM and EPF Composer are suggested.

Keywords

Quality Attribute Techniques, SPEM, EPF Composer
1 Introduction

The quality of software products need to be developed and assured throughout the development process [1, p.27]. According to [2, p.60], software product quality is defined as ``the degree to which a system, component, or process meets specified requirements, customer or user needs or expectations''. Examples of product qualities include safety, performance, and security. Specific techniques are available to achieve product quality in the development of critical systems. In this paper, we call these techniques ``Quality Attribute Techniques''(QAT). Examples of QAT for safety include hazard analysis techniques such as Failure Mode and Effect Analysis (FMEA) and Fault Tree Analysis (FTA).

QAT are used in software development in order to achieve required levels of product quality requirements such as safety and performance. There are many complementary and alternative [3] and it can be difficult to choose the most appropriate techniques. There are some existing guidelines and approaches to help development teams to meet specific product qualities. Examples for safety-critical software systems are [4-8]. Most software processes tailoring methodologies are designed to address issues such as project characteristics, size of the organisations, standard compliance in process tailoring and the technical and social suitability of the organization [9-11] QAT are used but not represented in detail or not incorporated well in software development process models [12, 13]. These guidelines and approaches usually target one specific group of QAT and may not be appropriate to be applied for other QAT. There have been no general approaches proposed for QAT representation and integration with software processes. It is difficult to communicate, monitor and analyse the relationship between QATs and other process elements if QATs are not well presented in process models. A good representation that allows effective ``change” mechanisms is also needed to support process tailoring, by facilitating the selection of QAT for inclusion into process models that target specific product qualities.

A number of metamodels such as Software Process Engineering Metamodel (SPEM) [14,15] OPEN Process Framework (OPF) [16], OOSPICE metamodel and Standard Metamodel for Software Development Methodologies (SMSDM) [17] have been developed to specify the concepts, rules and relationships used to define software development processes and their components. In this paper, SPEM has been selected to represent the main characteristics of QAT and the relationship of QAT with other process elements. EPF Composer is a process modelling tool based on the SPEM metamodel and is used in this paper to define QAT and incorporate QAT into the OpenUP process model.

The following two research questions are the focus of this study:

- How can QAT be represented using EPF Composer and the SPEM metamodel?
- What are the tailoring features in EPF Composer that can be used to integrate QAT into software process models?

This paper proposes and evaluates two approaches for representing QAT for better integration of QAT with process models in SPEM and EPF Composer. Safety techniques have been selected as a case example for evaluation. The outline of this report is as follows. Section 2 discusses QAT, SPEM and EPF Composer. Section 3 describes the two alternative representations of QAT. Section 4 discusses the evaluation of the two alternatives. Section 5 discusses the evaluation and suggests some improvements to SPEM and EPF Composer. Section 6 presents the conclusion of this study and discusses future research.

2 Background and Overview
2.1 Quality Attribute Techniques

QAT are techniques that are used to identify, eliminate, reduce, control and minimise potential quality problems in the development of critical systems. For examples, safety critical systems are concerned with a measure of the hazards to life, property or environment, and security-critical systems focuses on resistance to external threats and malicious actions against its integrity [3] and performance-critical systems emphasize response time or throughput [18]. However, QAT are not represented in detail in software development process models. Also, QAT are not well integrated with other process elements such as tasks, roles and work products in different stages of a process model.

QAT for safety-critical system have been selected as a case for initial evaluation in this paper because the area of system safety is well-established. There are many existing procedures, handbooks, standards, books and other references. In safety-critical systems, techniques are available to perform hazard evaluation, control and analysis. A hazard is a state or set of conditions of a system that, together with other conditions in the environment of the system, can lead to an accident or loss event. [3]. Safety techniques can be grouped into different categories such as hazard identification, hazard analysis and safety testing.

An initial investigation has been conducted in safety area to capture characteristics of QAT. Some important characteristics which provide detailed information of QAT and crucial to integrate QAT into process models have been extracted from the literature: [3,19-22]. An example of using these characteristics to describe the FMEA technique is shown below:

- **Technique Name**: Failure Modes and Effects Analysis (FMEA)
- **Aims**: Primarily used for subsystem and system hazard analyses. It also helps to identify critical items in terms of safety and reliability.
- **Description**: FMEA analyses which failures in a system can lead to undesirable situations. The probability and seriousness of the results for each failure mode are calculated. Corrective actions are prioritised and recommended.
- **Main performer(s)**: Safety Engineer
- **Optional Performer(s)**: System Analyst, Design Engineer, End User, Project Manager
- **Step(s)**: 1. Define the system to be analysed and determine the scope of the analysis. 2. Analyse and organise potential failure modes and their effects. 3. Identify and prioritise corrective action.
- **Input**: Design drawings, functional diagrams, previous analytical data, system descriptions, lessons-learned data, PHL, PHA report
- **Output**: FMEA worksheets, FMEA reports, Critical Item List (CIL)
- **Guidance Documents**: FMEA Worksheet Template, FMEA Guideline
- **As Source Data for (optional)**: Fault Tree Analysis (FTA)
- **Category**: Hazard Analysis
- **Benefits**: Thorough and systematic approach, quickly reveals critical single-point, reliability can be evaluated in detail
- **Limitations**: Does not consider system effects or human error, detailed and expensive to apply to large systems
- **Cost of Application**: Moderate (week)
- **Expertise**: Moderate training (strongly dependent on analyst's understanding of the failure modes)
- **Phase(s)**: Subsystem and system hazard analysis during design phase
- **Team/Individual approach**: Team
- **Single/Multiple Failures Analysis**: Single
Two alternatives are shown in section 3 to represent these important characteristics of QAT and incorporate them into process models.

### 2.2 SPEM Metamodel

SPEM is a metamodel for defining processes and their components using UML as a concrete notation [14,15]. Fig. 1 illustrates parts of SPEM metamodel.

![SPEM Metamodel Diagram](image)

**Figure 1:** SPEM Metamodel (adapted from [14,17]). `ModelElement` is a super-class of the other classes in this diagram.

The core idea of SPEM is that a development process is a collaboration of multiple process elements to achieve a specific project goal. A development process lifecycle can be structured into phases and iterations. Work definition describes the activity (i.e. tasks, operations and actions) performed in a process by a process role. An activity can be subdivided into steps. Work products or artefacts (e.g. a piece of code, a document, a mode and source code) are produced, consumed, or modified when multiple roles interact or collaborate during the process execution. Process components called Disciplines are used to categorise activities which share a common theme. Each activity is grouped under a Discipline. A Guidance element can be attached to any of the process elements shown in Fig. 1 (e.g. Activity, Step and Process Role). More detailed information of a process element can be provided by Guidance elements (e.g. checklist, guideline, practice, report and template).

SPEM has been selected to represent the important characteristics of QAT and integrate QAT into process models. Section 3 discusses two alternative representations of QAT based on the SPEM. The ability of SPEM in supporting representation and integration for QAT are evaluated in section 4.

### 2.3 EPF Composer

EPF Composer is a process modelling tool platform and extensible conceptual framework based on SPEM [15] for authoring, maintaining and customising software development processes [23]. Following SPEM, reusable method content is defined separately from its use in processes in EPF Composer. EPF Composer also provides three sample process frameworks, i.e. OpenUP/Basic, Extreme Programming and Scrum. Users can choose and customise existing process frameworks or
create new ones. These frameworks can be transformed into Electronic Process Guides (EPGs). This provides convenient access and assists navigation of the process model for process performers.

In section 3, EPF Composer is used to define two alternative representations based on SPEM for QAT and incorporate QAT into process models. OpenUP/Basic has been selected as an example of process model in this paper. The evaluation of the ability of EPF Composers to support representation and integration of QAT is discussed in section 4.

3 Representation of QAT Using SPEM Metamodel & EPF Composer

3.1 Overview

Two alternatives are shown in this section to represent important characteristics of QAT and integrate them into process models. The first alternative represents a QAT as a Step and the second alternative represents a QAT as a Task. These two process elements have been selected to represent QAT because the important characteristics of QAT can be represented in more structured formats. Additionally, the relationship between QAT and other process elements can be defined clearly. Guidance element and its supported Guidance Kind are less structured to represent important characteristics for QAT. As a result, Guidance is only used to describe additional information for QAT. The process element Activity in SPEM (see Fig.1) is similar to the process element Task in EPF Composer. A Task consists of a number of Steps. Safety techniques are used as case example for these two alternatives.

For both alternatives, a new method plug-in that based on OpenUP is defined to customise method content without directly changing the original content of the OpenUP. It comprises new method content, Disciplines, and Capability Patterns to represent new safety processes. The new components will be reusable for new releases of OpenUP plug-in. Four kinds of Content Variability mechanisms allow process engineers to create new plug-in that contribute, extend, replace or extend-replace new definitions to the tasks in the original plug-in [23].

3.2 Alternative 1: Step-Based Approach

In the Step-Based Approach, a QAT is represented as a Step in a Task. The main characteristics of QAT are added in the the step's description in EPF Composer. Alternatively, information of QAT can be described with Guidance documents such as guidelines, templates and worksheets. Hyperlinks to these Guidance documents need to be added to description editor for QAT Step. If another Task also use the same technique, manual duplication of the content is required by adding new Step and attaching Guidance documents to this Task. Step cannot be reused in other Task. This approach aimed to incorporate QAT as a step into a task in process models. QAT can also be added as a Step in new Tasks added to achieve specific product quality goal such as safety and security.

Safety techniques are used as case example for Step-Based Approach. New content packages such as risk assessment, risk control and general safety have been defined to categorise and maintain safety tasks. Under each content package, new safety Tasks such as Preliminary Hazard Identification (PHI), Preliminary Hazard Analysis (PHA), Subsystem Hazard Analysis (SSHA) and System Hazard Analysis (SHA) are created to represent safety-critical project specific processes. Safety techniques are added as Steps in appropriate safety Tasks. Important characteristics captured for each safety technique can be added manually to step's description or provided by using guidance documents such as guidelines, examples and templates.

3.2.1 Tailoring Feature(s) Used

To integrate QAT into process models, the new QAT Step is attached to one Task. Content Variability
mechanism "contribute" can be used to append a new QAT step to the original content of an existing task without changing original content. The sequence of this step can be changed in the parent task to show the most appropriate order to perform this technique. Disciplines have used to categorise Tasks that share the common quality goal and use QAT. A new software development lifecycle have been defined to include new QAT Step. Three options of process variability are available to copy these processes: Copy, Extend or Deep Copy. "Deep Copy" function has been used because the sequence of these tasks can be reorganised after selecting them into Work Breakdown Structure (WBS). As a case example, five "Disciplines" are used to group safety project specific processes: hazard analysis, safety coding, safety design, safety review and safety test. A new software development lifecycle, "SafetyProcessLifecycle1" has been defined to include existing OpenUP process and also appropriate safety tasks from the five disciplines.

3.2.2 Process view

A new project configuration has been defined to build new process without making changes directly to the original plug-in. A custom category has been added to generate a new process view to customise the process content and methods to be published. For the case example, new safety plug-in and Open-UP plug-in was added to this configuration. SafetyProcessLifecycle1, safety disciplines and safety roles have been selected to populate navigation views and new EPG for safety related projects.

3.3 Alternative 2: Task-Based Approach

In the Task-Based Approach, a QAT is represented as a Task that describes what needs to be performed by different roles and what work products are used as input and output to achieve a specific quality goal. The main characteristics of QAT is added in different attributes of a Task in EPF Composer. Additional information of QAT can be provided by using Guidance documents such as worksheets, templates, guidelines and examples. This approach aims to incorporate QAT as a new Task in process models. Alternatively, new QAT Tasks can extend, contribute or replace original content of existing task in process models.

As a case example, new Tasks are created to represent different safety techniques such as FMEA and FTA. New content packages such as risk assessment, risk control and general safety have been defined to categorise and maintain safety technique's Tasks.

3.3.1 Tailoring Feature(s) Used

Two ways have been used to integrate QATs Task to processes. The first is using Content Variability mechanisms to "contribute", "extend" or "replace" QAT information to a base task in process models to avoid modifying the original content. Both content from contributing QAT Task and base Task will become unified in new browsing view. As a case example, "contribute" or "replace" mechanisms have been used to respectively append or replace the new content of a QAT task to the original OpenUP content.

Another way of integrating a QAT Task to process models is using Capability Patterns. These Process Patterns can be used to categorise different types of QAT Tasks. Additionally, Capability Patterns allow process engineers to indicate the flow of tasks and create activity diagrams for project specific processes. They can be reuse and applied to many different lifecycle processes. As a case example, six Process Patterns have been defined for safety technique Tasks: Preliminary Hazard Analysis, Safety Design, Safety Testing, SSHA and SHA. A new software development lifecycle, "SafetyProcessLifecycle2" has been defined to include appropriate process patterns, safety technique's Tasks and existing process content from OpenUP plug-in.

3.3.2 Process view

In the approach, a new project configuration is defined for new processes without directly changing the original plug-in. A custom category is added to generate a new process view to customise the process content and published methods. In the safety example, a new safety plug-in and Open-UP plug were
4 Evaluation

This section evaluates the capability of SPEM metamodel and EPF Composer in supporting two alternative representations shown in previous section. The evaluation identifies advantages and limitations of the SPEM and EPF Composer.

4.1 Evaluation Criteria

Four criteria have been selected for the evaluation based on the criteria for effective process models defined by Humphrey and Kellner [24]. They highlight that, “process models must represent the way the work is actually (or is to be) performed, provide a flexible and easily understandable, yet powerful, framework for representing and enhancing the process and be refirable to whatever level of detail is needed” [24, p.332]. For the purpose of this studies, we focus on how the two alternatives represent characteristics of QAT, how easy to integrate QAT into process models, the reusability of process elements and contents defined by the approach and maintainability to manage the representation and integration.

- **Representation**: How well the important characteristics of a QAT can be captured with the approach?
- **Ease of Integration**: Is it easy to integrate QAT to a development process model using the approach?
- **Reusability**: When using the approach, can the QAT be applied for different processes?
- **Maintainability**: How convenient is it to manage the representation and integration using the approach?

4.2 Evaluation for Step-Based Approach

- **Representation**: A QAT can be easily added as a simple *Step* in a *Task*. The main characteristics of each QAT can only be added manually in a *Step* description or represented as guidance documents such as guidelines and templates. These guidelines and templates can be attached to the *Task* or put as hyperlinks in the description of the *Step* created to perform this QAT. There is no specific format for the guideline, any information can be entered for a specific QAT. The approach fails to show the relationship of this QAT with other process elements like *Roles* and *Work Products*. Users are not able to have a clear understanding about who will perform this QAT and what kind of input and output artifacts are required for this QAT.

- **Ease of Integration**: Adding *Step* to a *Task* is a simple way to integrate a QAT to a safety *Task* or an existing OpenUP *Task*. If a *Task* needs two different QAT, an additional *Step* to present the second QAT can be added easily to this *Task*. However, duplicate copies of the *Step* need to be created manually for multiple *Tasks* which use a same technique. Additionally, *Content Variability* mechanisms do not allow users to “contribute” a new QAT *Step* to more than one existing *Task* in process models. It is also difficult to show the relationships of a QAT with other process elements.

- **Reusability**: New safety *Tasks* can be reused but a *Step* which has been used to integrate a safety technique with a *Task* cannot be shared with other *Tasks*. User must manually duplicate the *Step* and link or attach guidance documents to every *Task* which requires the same technique. Alternatively, user can reuse the same technique by attaching *Guidance* documents of this technique to any *Task* without adding a *Step*. However, some *Guidance* kinds such as template, roadmap, practice and term definition cannot be added to a *Task* in EPF Composer. Additionally,
Content Variability mechanisms which are used to `contribute` new QAT Step to an existing Task in process models do not allow users to reuse same QAT to multiple Tasks.

- **Maintainability**: New safety Tasks such as hazard analysis has been defined to represent safety-critical project specific processes when no generic OpenUP Task is relevant to the QAT Step. Extension or contribution can be done to existing Tasks or new safety Tasks. It is easier to maintain the project specific Tasks by adding extra Tasks in new method plug-in.

### 4.3 Evaluation for Task-Based Approach

- **Representation**: Most of the main characteristics of a QAT can be represented well in a Task. Attributes in a Task allow process engineers to include detailed information for the QAT such as purpose, main description and steps to conduct this QAT. Additionally, these QAT were able to relate with other process elements such as Work Products and Roles. A clear relationship can be shown. Process performers are able to find out who will perform this QAT and what kind of input and output artifacts are required for this QAT. However, when a QAT Task contributes to an existing Task, some of the important information of this QAT like presentation name will not be shown. Process performers may not be able to differentiate the main information of a QAT from the original content of an OpenUP or a safety lifecycle Task. Additionally, if more than one QAT contribute to a Task; the content of this Task will become more complicated.

- **Ease of Integration**: The first way of integration a QAT Task to an existing Task by contributing, extending or replacing a QAT description to an existing task. However, a QAT Task cannot contribute, extend or replace multiple existing tasks. A more flexible method for integration is using Process Patterns to include QAT Tasks and OpenUP Tasks. A QAT Task can be applied to different Process Patterns which require the same technique.

- **Reusability**: The first way of integration a QAT Task to an existing Task by contributing, extending or replacing a QAT description to an existing task. However, a QAT Task cannot contribute, extend or replace multiple existing tasks. A more flexible method for integration is using Process Patterns to include QAT Tasks and OpenUP Tasks. A QAT Task can be applied to different Process Patterns which require the same technique.

- **Maintainability**: Another configuration needs to be defined in order to show the Process Patterns, QAT Tasks and contributed content of a safety technique. This is because the main configuration will only show the original content of the process element. This is a way to avoid overriding the original content of an element.

### 5 Discussion

Both alternatives have their own advantages and limitations. The Step-Based Approach is a structured and simple way to integrate a QAT to an existing Task in a process model. However, adding a Step is hard to highlight the main characteristics of a QAT since all the information need to be added as links to guidance documents or entered manually in the Step's description. Additionally, relationships of QAT with other process elements such as work products and roles are harder to recognise since they were added as hyperlinks in the descriptions of the Step. The Task-Based Approach which uses a Task to show the characteristics of QAT is a more structured way to represent QAT. Relationships of QAT and other process elements such as Work Products and Process Roles are able to be modelled clearly in EPF Composer. Process Patterns are more useful to integrate and categorise QAT in different delivery processes. A Process Pattern can be reused in many parts in a delivery process with individual customisation (e.g. remove unnecessary tasks or add new tasks) for the pattern's content.

Based on the evaluation, the main limitation for both alternatives is that some problems exists when manual duplication is required to reuse the same technique for different processes. Process engineers can copy whole or parts of existing processes from a method content library for duplication. However, some of the content (e.g. relationship with other process element) which are inherited from original method contents cannot be modified. Content Variability mechanisms only can contribute, extend or
replace content of QAT to a Task in EPF Composer. Contribution, extension or replacement of QAT’s Step or Task to multiple Tasks in process models are not allowed.

The basic concept of SPEM regarding clear separation of method content from how it is used in processes is useful for the integration of QAT into software development process models. This provides flexibility to incorporate QAT Tasks or Steps into any part of process models. There are some limitations of existing SPEM metamodel and EPF Composer to support the representation of a QAT and also to incorporate it into process models. Selection strategy and tailoring method for more effective QAT selection and integration with software processes are out of the scope of this paper. The following suggestions may improve this:

- Each Step in a Task requires different Performers, contribute different Work Products to a task and also need different Guidance elements to provide more detailed information to practitioners. SPEM can be extended to allow process engineers to assign Performer(s), Work products and Guidance elements to a specific Step in a Task in a more structured way.

- A new type of GuidanceKind can be added in SPEM for QAT. It can include attributes to represent important characteristics of QAT in a more structured way. This new GuidanceKind should be able to attached to multiple Tasks or Steps that using the same QAT in processes.

- Multiple tasks or steps may use the same technique. The Content Variability mechanisms in SPEM can be extended to support contribution, extension or replacement of a QAT’s Task or Step to multiple existing Tasks.

- New attributes can be added to the Task element in EPF Composer to represent some important characteristics of QAT.

- EPF Composer can provide a more flexible approach to support the improvements of SPEM, reusability of a QAT and at the same time provide strong capability to maintain the consistency of process models after any change.

- Besides the important characteristics of QAT, Guidance documents such as guidelines, templates and examples can be used to provide additional information for practitioners to execute the QAT efficiently and accurately.

### 6 Conclusion and Future Work

The goal of using QAT is to identify, eliminate, reduce, control and minimise potential quality problems in the development of critical systems. QAT are used to improve product qualities such as safety, performance and security. However, QAT are not well integrated into software development process models. A good representation allows development teams to communicate, monitor, control and analyse the integration of QATs with software processes effectively. This study has investigated how the SPEM metamodel and EPF Composer can support the representation of characteristics of QAT for better integration into software process models such as OpenUP. Two alternatives - a Step-Based Approach and a Task-Based Approach have been proposed for the representation of QAT and integration with software process models. Safety techniques were selected as a case example for an evaluation. Based on the evaluation, there are several benefits and limitations of the SPEM metamodel and EPF Composer in supporting the two alternatives. The concept of separation of method content from processes in SPEM is useful to define characteristics of QAT and incorporate QAT into different processes in process models. However, there are still some limitations especially when the process content of QAT are manually duplicated to be reused for different processes. Some content inherited from the original method content cannot be modified.

Process models which are defined based on metamodels are expected to be more extensible, reusable, configurable and tailorable to specific project needs. Some of the future challenges of metamodels and process modelling tools are to include providing improved support for process component reuse, representation of QAT and integration of a process element into process models. Future work of this research are developing a tailoring method to incorporate appropriate QAT into software development process models. SPEM metamodel and EPF Composer will be extended to support the integration approach.
7 Literature

8 Author CVs

Yin Kia Chiam

Yin Kia, Chiam is a PHD student of the Computer Science and Engineering Department at the University of New South Wales. She is also a lecturer of the Computer Science department at the University of Malaya. Before pursuing her PHD, Yin Kia had position as a Software Test Engineer in Motorola Malaysia Software Center. She holds a MSc in Information Technology from Malaysia University of Science and Technology (MUST), which is a collaborative programme between MUST Ehsan Foundation and the Massachusetts Institute of Technology (MIT).

Mark Staples

Dr. Mark Staples is a Research Group Manager at NICTA, Australia's ICT Centre of Excellence. Before joining NICTA, Mark had positions in the software engineering industry as a Process and Tools Manager at a medium-sized enterprise software development company, and as a Configuration Manager and Test Engineer in the development of the safety-critical environmental control system of a mass transit system. He is a Conjoint Senior Lecturer at the University of New South Wales, and holds a PhD from the University of Cambridge.

Liming Zhu

Dr. Liming Zhu is a researcher in the Software Engineering group at National ICT Australia (NICTA) and teaches software engineering at University of New South Wales. He formerly worked in several technology lead positions in software industry before pursuing a PHD degree in software engineering at University of New South Wales (UNSW). He has worked with Australia government agencies, defense, standardization bodies and commercial companies. He is a committee member of the Standards Australia IT -015 (system and software engineering) group. He contributes to the ISO/SC7 Working Group 42 on architecture related ISO standards. His research interests include software architecture, service engineering, model driven development, business process modeling and software development processes.

Acknowledgements

NICTA is funded by the Australian Government as represented by the Department of Broadband, Communications and the Digital Economy and the Australian Research Council through the ICT Centre of Excellence program.