Software Process Improvement Initiatives Based on Quality Assurance Strategies: A QATAM Pilot Application

Dietmar Winkler[1], Frank Elberzhager[2], Stefan Biffl[1], and Robert Eschbach[2]

[1] Institute of Software Technology and Interactive Systems,
Vienna University of Technology,
Favoritenstrasse 9/188, 1040 Vienna, Austria
{Dietmar.Winkler,Stefan.Biffl}@tuwien.ac.at
[2] Fraunhofer Institute of Experimental Software Engineering
Fraunhofer-Platz 1, 67663 Kaiserslautern, Germany
{Frank.Elberzhager,Robert.Eschbach}@iese.fraunhofer.de

Abstract. Quality Assurance (QA) strategies, i.e., bundles of verification and validation approaches embedded within a balanced software process can support project and quality managers in systematically planning and implementing improvement initiatives. New and modified processes and methods come up frequently that seems promising candidates for improvement. Nevertheless, the impact of processes and methods strongly depends on individual project contexts. A major challenge is how to systematically select and implement "best-practices" for product construction, verification, and validation. In this paper we present the Quality Assurance Tradeoff Analysis Method (QATAM) that supports engineers in (a) systematically identifying candidate QA strategies and (b) evaluating QA strategy variants in a given project context. We evaluate feasibility and usefulness in a pilot application in a medium-size software engineering organization. Main results were that QATAM was considered useful for identifying and evaluating various improvement initiatives applicable for large organizations as well as for small and medium enterprises.

Keywords: Quality Assurance, Quality Assurance Strategies, Strategy evaluation, Best-Practices Method Selection.

1 Introduction

Software process improvement initiatives are usually implemented within a Quality Management System (QMS) based on standards, like ISO 9001 [12], CMM(I)[1], and ISO/IEC 15504 (SPICE) [13] to (a) enable compliance to these standards and maturity level concepts and (b) to enable, ensure, and improve project, process, and product quality. A QMS aims at providing a basic framework for improvement initiatives and need effective and efficient processes and methods for project application. Successful

[1] CMMI: http://www.sei.cmu.edu/cmmi/

development projects require (a) appropriate project organization approaches (e.g., software processes), (b) methods for product construction, and (c) methods for verification and validation (V&V). Typically, improvement initiatives address individual aspects of software development.

Following Deming's Plan-Do-Check-Act Cycle (PDCA-Cycle) [6], improvement initiatives focus on selection and modification of individual project and process aspects *(Plan)*, implementation of changes *(Do)*, evaluation of the impact of modifications on the project progress and product quality *(Check)*, and ascription of evaluation results to the next planning stage *(Act)*. An ongoing application of the PDCA-Cycle leads to continuous improvement of processes and products. Nevertheless, improvement strategies need additional resources for implementation and application. A lack of resources can hinder improvement initiatives in small and medium organizations (SME). Thus, efficient approaches for QA strategy selection and evaluation can also support SMEs in initiating process and product improvement initiatives. Our observation in software and systems development industry showed two promising candidate approaches for improvements on project and product level.

- *Selection and application support for processes and methods.* Selecting appropriate processes and methods in a given project context is an ongoing challenge in the software engineering industry and strongly depends on project attributes and individual experiences of project and quality managers within an organization [4]. Efficient selection of best-practice processes and methods is a key challenge.
- *Bundling of methods to increase project performance.* Observations in industry showed that isolated method application lead to an isolated improvement of individual artifacts. Additional benefits can arise from bundling methods across phases (and disciplines) due to synergy effects and raise overall project performance [3], e.g., results from software inspection can foster test case generation.

The *Quality Assurance Tradeoff Analysis Method* (QATAM) provides systematic support for (a) selecting best-practice processes and (bundles of) methods and (b) enables the evaluation of candidate QA strategies for project application [3]. In this paper we present the QATAM concept to support the selection of efficient QA methods with respect to various project and organization context parameters: (a) large organizations can benefit from considering company-specific and common best-practices and (b) small and medium enterprises can benefit from selecting "light-weight" approaches in a given context with limited resources. Thus, QATAM aims at supporting (a) the initiation of systematic improvement initiatives for products and processes, (b) the identification of comprehensive QA strategies in a given context, and (c) the systematic evaluation of QA strategy variants with respect to an effective and efficient implementation of these strategies [3][15]. We evaluate the concept of QATAM in a pilot study in a medium-size software development organization [15].

The remainder of this paper is structured as follows: Section 2 describes success factors of improvement initiatives and related work on processes and methods for QA strategy development. Section 3 describes the research issues. We present the QATAM process approach in section 4 and the evaluation study in section 5. Finally, section 6 concludes and identifies future work.

2 Success Factors and Approaches for Process Improvement

Initiating improvement initiatives require (a) systematic processes and (b) defined success criteria to evaluate improvement activities. Various studies report on the identification of key success factors of improvement initiatives based on measurement [9] and maturity-based analysis approaches [17]. Niazi *et al.* present a maturity model for SPI based on the CMMI process model [16]. Dybä reports on the results of an empirical investigation on key success factors in a quantitative survey of 120 software organizations [10]. Stelzer *et al.* analyze success factors of organizational changes in software process improvement based on CMM(I) and ISO 9001 and came up with ten common key factors [20], derived from literature and analyzing 56 software organizations (Table 1). We focus on *management support*, *staff involvement* and *tailoring improvement initiatives* to *set relevant and realistic* objectives as most important impact factors of QATAM in the decision and planning phase of improvement initiatives.

Table 1. Success Factors according to Stelzer *et al.* [20]

Rank	Success Factor	Rank	Success Factor
1	*Management support*	6	Change agents and opinion leaders
2	*Staff involvement*	7	Stabilizing changed processes
3	Providing enhanced understanding	8	Communication & collaboration
4	*Tailoring improvement initiatives*	*9*	*Set relevant and realistic objectives*
5	Managing the improvement project	10	Unfreezing the organization

In general, improvement initiatives are based on two important aspects: (a) *what* should be improved and (b) *how* this improvement approach should be implemented and evaluated. Improving projects and products strongly depend on the application of efficient and effective methods aligned with a well-structured software process. Software processes typically consist of a sequence of steps, separated by milestones (or decision gates) [4]. Defined measures of product and quality attributes can be used for product quality estimation and project control. For instance, defect estimation methods use defect detection data from software inspection to predict the number of remaining defects in the artifact under inspection [5]. Based on these results project managers can decide to introduce additional QA activities or to proceed with the next step of product development. Measurement and feedback on product, process and project results are the foundation for product and process improvement on various levels as illustrated by the PDCA-Cycle [6]. Typically, improvement cycles include four steps: (a) definition of goals (planning stage), (b) execution of a defined task, (c) V&V of the results using qualitative and quantitative data, and (d) feedback to improve products, processes, and projects in follow-up iterations. Systematic application of improvement initiatives leads to an overall increased product, process and project quality.

Furthermore, collected knowledge can be seen as the foundation for experience bases [1] within learning organizations [18]. Experience bases with empirical data provide specific knowledge on future application in comparable contexts. (Expert) experience and empirical knowledge on method application is the foundation for the quality assurance tradeoff analysis method (QATAM) [3].

2.1 Instruments for Improvement Initiatives: Software Processes and Methods

Traditional and sequential software processes (e.g., V-Model XT [4]) and agile process models (e.g., Scrum [19]) are wide-spread processes for planning and controlling software engineering projects. These processes provide defined sequences of steps for project execution and can facilitate improvement initiatives. Nevertheless, selecting and tailoring an appropriate software process strongly depends on project attributes, e.g., project size (e.g., number of involved project members and duration), project type, application domain, and volatility/stability of customer requirements [4]. Typically, project phases, separated by decision gates, include isolated constructive approaches, e.g., test-driven and/or model-driven development, and V&V methods, e.g., inspection and testing, to assess deliverables for compliance with basic requirements and specification documents. Results from V&V method application are the foundation for product and process improvement.

Analytical quality assurance approaches, e.g., inspections and testing, support engineers in evaluating products, projects and processes. Data collected during method application provides quantitative and qualitative information on the artifact under investigation and enable decision making processes during project monitoring and controlling [18]. Software inspection – a formal quality assurance approach – focuses on defect detection in early phases of development and supports project and quality managers in project controlling, e.g., based on defect content estimation [5]. Software testing approaches focus on the identification of product deviations in late phases of software development based on executable code. Test-driven development (TDD) [2] – a constructive software development practice – shortens the duration of test case generation and execution and includes early defect detection by bundling test case generation, code implementation, and test execution. Measures from V&V can support project and quality managers in product assessment and improvement.

The application of individual methods includes isolated benefits in every individual stage of software development. Nevertheless, bundling various methods might include additional benefits because of synergy effects, e.g., higher overall product quality and reduced method application effort. For instance, test cases can be defined based on software inspection results (e.g., during software inspection) and can be applied on executable software code during traditional testing or TDD application.

2.2 Quality Assurances Strategies

Selecting and bundling individual QA activities lead to QA strategies [7]. QA strategies comprise relevant information to apply a bundle of quality assurance activities within a project. Additionally, a QA strategy includes related stakeholders, customized development processes, and available project information at a defined time, e.g., at a quality gate. Typically, a QA strategy is defined at project start by project and quality managers and adjusted during project execution based on the current state of the project, e.g., based V&V results. Thus, quality gates represent the minimum set of information for structuring a QA strategy (e.g., quality level of a certain product). For instance, quality gates at the end of requirements definition, design, coding, and testing phases can be used as roadmap for QA strategy planning.

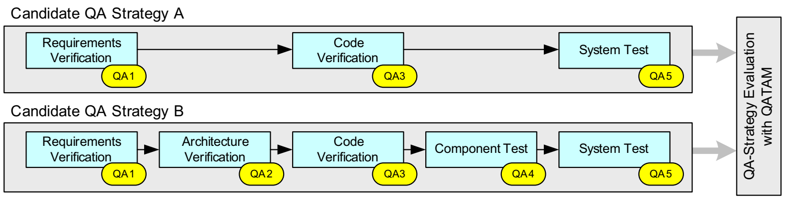

Fig. 1. Example of two different QA Strategies

Figure 1 presents two basic candidate QA strategy examples for a software project. Note that QA_x represents individual QA activities at defined quality gates. *QA strategy A* includes three basic QA steps, e.g., defined by company standards. This initial QA strategy typically represents the *As-It-Is strategy* of an organization and can be used as starting point for improvement initiatives. An alternative *QA strategy B* includes additional process steps, i.e., architecture verification and component tests. Both strategies are the input for QA strategy evaluation within QATAM.

Thus, main goals of QATAM are (a) to systematically capture candidate QA strategies and (b) to evaluate every identified strategy with respect to given company and/or project needs.

3 Research Issues

Continuous improvement aspects are usually core elements of Quality Management Standards captured by audits and assessments. In general, improvement initiatives aim at improving processes, projects, and products within an organization. Nevertheless, introducing improvement initiatives require defined processes (driven by a QMS) for selecting appropriate measures and implementing changes. Because of limitations in suggested and defined methods, we see the need to establish a well-defined approach for (a) selecting most promising approaches for process improvement and (b) implementing selected strategies in a given context. In this paper we focus on the definition of candidate QA strategies and the evaluation with respect to identify the most valuable QA strategy for project application and identified two major research issues:

- *RI-1. Identifying a systematic process approach to support QA strategy development and evaluation/selection.* The initialization of improvement initiatives strongly depends on company and project requirements and success factors (see Table 1). Thus, we see the need to establish a well-defined evaluation framework for identifying and evaluating candidate QA strategies.
- *RI-2. Assessment of QATAM in an industry context.* To evaluate the strengths of QATAM – as a strategy identification and evaluation approach with – respect to feasibility and usefulness, we applied QATAM in a medium-size software project within a software development organization [15].

4 The Quality Assurance Tradeoff Analysis Method (QATAM)

The main goal of the QATAM framework, embedded within a Life-Cycle QM approach [8][15], is to support project and quality managers in planning and evaluating sequences of QA activities, i.e., quality assurance strategies, along the project life-cycle. Major outcome of the QATAM application is a selected QA strategy – a bundle of best-practice QA methods – in a given project context.

The QATAM framework consists of four basic components, presented in Figure 2.

- *Context and Scope (1).* Organizational standards, project context, goals from related stakeholders, and process constraints can limit QA strategy definition and define process constraints for improvement initiatives.
- *QA Method Repositories (2)* include candidate individual QA activities, e.g., different inspection and testing approaches classified by empirical data [1] and/or company and project experience. Note that every QA activity is characterized by a set of related attributes, e.g., effort for implementation and application, scope, application domain, ease of use, usability, effectiveness, and efficiency.
- *QA Strategy Development (3).* Mapping of context/scope information and candidate individual QA methods lead to a set of candidate QA strategies. QATAM provides two application approaches: (a) *Out-of-the-Box strategy development*, if no current strategy is available or radically change is required and (b) *Step-by-step improvement* based on an established strategy. The latter refers to improvement initiatives in "small steps" as a typical use case in industry environment.
- *QA Strategy Evaluation & Selection (4).* Various candidate QA strategies, goals and scenarios, derived from various stakeholders are the input for the QATAM evaluation process and lead to a single "best-practice" QA strategy (or a prioritized list of a few QA strategies).

The major outcome of the QA strategy evaluation is a selected best-practice strategy compliant to organizational standards and the project context supported by all relevant stakeholders. Additionally, lessons learned from QATAM application can be reused to update the QA-method repository. Note that we focus on the improvement initiatives in this paper as a typical use case in industry environment (Figure 2, 3b).

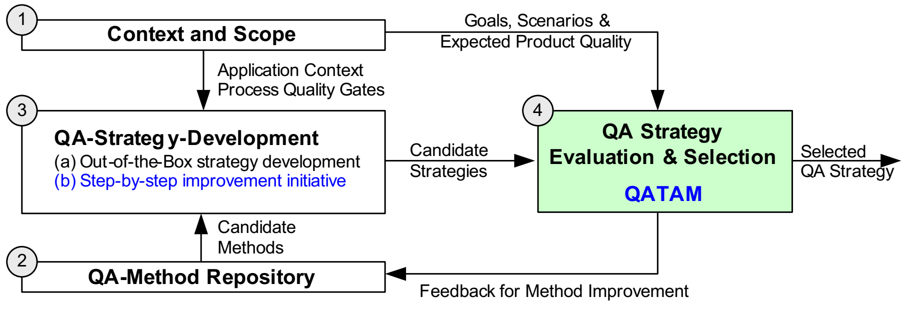

Fig. 2. The QATAM Framework

4.1 Improvement Initiatives with a "Step-by-Step"-Approach

In order to perform a step-wise improvement rather than radically change a complete QA strategy (which is typically not feasible in practice because of cost and resource constraints), the improvement initiative approach is based on the need to introduce "small steps of improvement" to achieve process, project, and product quality improvement.

A QA strategy development in an improvement initiative (Figure 2, 3b) includes three major steps: (a) *As-it-is analysis* to capture current engineering practices; (b) Improvement of *established practices*; and (c) Introduction of *novel practices*. Nevertheless, selecting appropriate improvement approaches requires information on the expected effectiveness and efficiency of candidate improvement strategies. Thus, measurement or expert knowledge can be used to estimate improvement capabilities, e.g., defect content estimation based on previous projects and/or empirical data. The main outcome of QA strategy development is a set of individual QA strategies for evaluation purposes.

4.2 Evaluation Process

The major goal of the QA strategy evaluation process is to select the most promising candidate strategy for implementation. Table 2 summarizes the basic QATAM steps, adapted from the Architecture Tradeoff Analysis Method (ATAM) [14], a well-investigated approach for architecture evaluation. Note that Table 2 includes the basic steps, as well as input information and deliverables.

Table 2. QATAM steps including input and output information

Step	Input	Output
1. QA Strategy Development	Current practices; Involved stakeholders	List of comprehensive Candidate QA strategies
2. Scenario Brainstorming	Context information	Set of scenarios (grouped by stakeholder groups)
3. Pre-selection of possible QA strategies	Set of QA strategies	Refined set of QA strategies
4. Determination of scenario coverage	Refined set of QA strategies & set of grouped scenarios	Estimated scenario coverage regarding each QA strategy
5. Prioritization of scenarios regarding risk and relevance	Set of grouped scenarios	Prioritized set of grouped scenarios
6. Evaluation of QA strategies	Refined set of QA strategies & prioritized set of grouped scenarios	Evaluated QA strategies regarding stakeholder scenario groups and prioritized scenarios
7. Determination of success factors	Refined set of QA strategies & relevant scenario(s)	Analyzed QA strategies regarding determined success factors of relevant scenario(s)
8. Trade-off analysis & determination of one "best-practice" QA strategy.	Refined set of QA strategies, Results of strategy evaluation, results of success factor analysis	One best-practice QA strategy

5 The QATAM Pilot Application Case Study

Preliminary steps of the QATAM application pilot application focus on (a) context information collection and (b) the identification of candidate improvement initiatives based on an as-it-is process approach and the development of candidate alternative strategies based on empirical data and experience. Context is a medium-sized software development company (see details on the case study organization in [15]). Goal is the improvement of change request handling, quality improvement of individual deliverables, high user acceptance of process modifications, and increasing product development efficiency and effectiveness.

Step 1: Planning of QA Strategy Evaluation. The first step of QATAM includes the identification of the current software process, involved stakeholders, and alternative QA strategies. Basically the *current development process* consists of three major steps, comparable to a simplified V-model approach: (a) analysis and prioritization of requirements and change requests, (b) implementation of changes including developer self-tests, and (c) automated and manual tests for verification and validation. Note that the as-it-is approach is used as reference candidate strategy for QATAM application (CS1). *Relevant stakeholders* are customers, developers and the management.

Alternative candidate QA strategies are:

- *CS2:* Additional reviews to address incomplete requirements and improve early defect detection activities in requirements definitions, use cases, and specification documents (method change).
- *CS3:* Introduction of additional software testing steps (i.e., unit and component tests) and reduction of developer self-tests (method change).
- *CS4:* Implementation of a tailored V-Modell XT process approach [4] based on the current V-model practices (process change).
- *CS5:* Introduction of Scrum as an agile process approach to improve change request handling and increased customer interaction [19] (process change).

Step 2: Scenario Brainstorming: Goals and scenarios can help focusing on individual application areas and issues addressed by the QA strategy [7]. We applied the EasyWinWin (EWW) approach [11], adapted from requirements elicitation processes to identify and prioritize individual goals and scenarios from various perspectives. This step includes a brainstorming session of all involved stakeholders.

Table 3. Snapshot of EWW Goal/Scenario Elicitation Workshop

No	Stakeholder	Goals/Scenario	Priority	Measurement
C1	Customer	Measurement of performance	B	Response time of web services
C2	Customer	Usability from end user view	A	Questionnaire for User Testing
D1	Developer	Change Request Handling	B	Process Assessment
D2	Developer	Frequent Changing Requirements	A	Number of change requests per time-unit
D3	Developer	Completeness & correctness of functional requirements	A	Number of defects detected
D4	Developer	Requirements traceability enabled	A	Coverage of requirements by test cases
M1	Management	Defect detection during development and test	A	Number of defects detected
M2	Management	Cost reduction & faster component development	A	Project tracking

The collected goals and scenarios were grouped and ordered sequentially. Table 3 presents a snapshot of the results of a EWW workshop to summarize goals and scenarios from different stakeholder perspectives. Note that we focus on goal/scenario D3 (completeness & correctness of functional requirements) for further evaluation.

Step 3: Pre-Selection of Candidate QA Strategies. The third step of QATAM includes an initial pre-selection of candidate quality assurance strategies to (a) exclude irrelevant initial candidate strategies, i.e., strategies which might not be reasonable in the given context, and (b) exclude scenarios which are rejected by the management of the organization to provide compliance to common organization standards, business goals, and management needs. Following this constraints and intensive discussions with the management of the organization, scenarios CS1, CS2 and CS5 were selected for further evaluation. Table 4 also presents the results of the selection process. Note again that CS1 represents the as-it-is strategy.

Table 4. Candidate QA Strategies and Pre-selected decisions

	Strategy	Change category	Comments	Decision	Selected
CS1	As-it-is	No change	No improvement of the current process	Evaluation	Yes
CS2	Additional Reviews	Method	Additional effort for implementation	Evaluation	Yes
CS3	Increased Test Effort	Method	Developer self-test are mandatory (management guidelines)	No option	No
CS4	V-Modell	Process	High implementation effort	No option	No
CS5	Agile Approach	Process	Short iterations and fast response times	Evaluation	Yes

Step 4: Scenario Coverage of Candidate QA Strategies. Based on the pre-selection of strategies and the identified scenarios, the next step focuses on the identification scenario coverage by the individual strategy. Coverage ranges from 0% (if the scenario is not affected by the strategy) to 100% coverage (scenario is fully covered). Typically this step is conducted by the key stakeholders in cooperation with process and method experts. An alternative approach of scenario coverage estimation can be based on empirical and historical data, if data of sufficient quality is available. Outcome of scenario coverage evaluation is an agreed list of scenarios and strategies and the expected contribution of every strategy to individual goals and scenarios. Table 5 presents a snapshot of the results of the candidate strategy coverage analysis.

Table 5. Coverage of Goals/Scenarios and Selected Candidate Strategies

No	Goals/Scenario	Priority	CS 1: As-it-is	CS 2: Reviews	CS 5: Agile
C1	Measurement of performance	B	50	50 (~)	50 (~)
C2	Usability from end user view	A	30	50 (+)	70 (+)
D1	Change Request Handling	B	80	80 (~)	70 (–)
D2	Frequently Changing Requirements	A	10	10 (~)	80 (+)
D3	**Completeness & correctness of functional requirements**	**A**	**20**	**70** (+)	**70** (+)
D4	Requirements traceability enabled	A	50	70 (+)	90 (+)
M1	Defect detection during development and test	A	30	80 (+)	80 (+)
M2	Cost reduction & faster component develop.	A	30	50 (+)	80 (+)

Step 5: Prioritization of Goals and Scenarios. Because of a large amount of goals and scenarios (even treated during the coverage analysis), not all of them might have a similar impact on project success. Thus, it is necessary to prioritize them regarding to their business value. Usually, this is done in a workshop including all involved stakeholders to identify the most important goals and scenarios. The major reason for prioritization after coverage evaluation is (a) to see the impact of all relevant

strategies on the goals and scenarios (to enable a comprehensive view) and (b) to support the prioritization process of the goals and scenarios with respect to the coverage. Table 3 and Table 5 also include the prioritized goals and scenarios derived from the prioritization workshop.

Step 6: Evaluation of QA Strategies. The aim of QA strategies evaluation is the identification of the most valuable QA strategy for implementation purposes (based on the highest coverage). As reported in the scenario brainstorming section (step 2) and prioritization (step 5), the collected scenarios and goals are classified according to stakeholders and priorities. The evaluation step focuses on this classification schema and evaluates estimation values according to their mean coverage (and expected potential improvement). Table 6 presents the evaluation results of classified high-priority goals and scenarios.

Table 6. Strategy Evaluation of Classified Scenarios

Goals/Scenarios	Priority	CS 1: As-it-is	CS 2: Reviews	CS 5: Agile
High Priority Goals/Scenarios	**A**	**28**	**55 (+)**	**78 (+)**
- Customer Scenarios	A	30	50 (+)	70 (+)
- Developer Scenarios	A	27	50 (+)	80 (+)
- Management Scenarios	A	30	65 (+)	80 (+)

Step 7: Determination of Success Factors. Defining and assessing success criteria depends on (a) expert knowledge derived from the key stakeholder group and (b) empirical data of method application in a given context. In the pilot study the key stakeholders estimated the success criteria for scenario application. Table 7 presents the results of success criteria evaluation in a decision workshop (derived from an EWW workshop [11]). We identified three major impact factors with respect to the application of candidate QA strategies in context of this pilot application:

- *Strategy Performance* refers to the effectiveness and efficiency of strategy application, e.g., number of identified defects per time-unit.
- *Effort of implementation and application* are key factors for management decisions to implement a strategy; high effort will hinder strategy implementation.
- *Impact of product quality on later phases*, e.g., during later development phases and/or during maintenance, was found a key success factor for strategy selection.

Note that we present the analysis results of success factors (see Table 7) with respect to goal/scenario D3 *"Completeness & correctness of functional requirements"*.

Step 8: Determination of a "Best-Practice QA Strategy". Based on prioritized goals and scenarios, scenario coverage, and impact and success factor evaluation, it is up to the team to draw appropriate conclusions and measures with respect to selecting the most promising QA strategy in the given context. Regarding the pilot application, the key stakeholder (and management) decided to follow strategy CS2 by including additional reviews within the current software development process (trade-off estimation of individual strategies). Nevertheless, the selected improvement step is the first step towards a continuous improvement strategy because a possible next step might be the implementation of agile software engineering practices according to CS5.

Table 7. Impact of Success Factors of Strategy Selection and Application

Goals/Scenario		CS 1: As-it-is	CS 2: Reviews	CS 5: Agile	Comments
Strategy Performance					
	- Effectiveness of defect detection	Low (–)	High (+)	High (+)	Ability to identify defects
	- Efficiency of defect detection	Low (–)	High (+)	High (+)	No. of defects per effort unit
Effort of implementation /application					
	- Effort (implementation)	n/a	Medium (+)	High (–)	Effort for strategy implementn.
	- Effort (application)	Low (+)	Medium (+)	Medium (+)	Effort for strategy application
Impact on later process phases					
	- Reduced defects in later phases	Low (–)	Medium (+)	Medium (+)	Expected benefits during development …
	- Reduced customer bug reports	Low (–)	High (+)	High (+)	.. and maintenance

6 Conclusion and Future Work

This paper presented the concepts of the QATAM approach, a framework for supporting project and quality managers in defining and evaluating QA strategies in a certain development context. Based on the well-established ATAM process approach for architecture evaluation [14], QATAM focuses on QA strategies with respect to improvement initiatives based on related benefits, risks, and cost of individual strategies. The process and the individual process steps were found feasible and useful by all involved stakeholders and can drive goal/scenario brainstorming, QA strategy development and evaluation of most value scenarios.

Lessons Learned. During evaluation of QATAM we derived a set of experiences, which may support practitioners in conducting QA strategy evaluation: (a) *Limitation of the number of „basic and candidate QA strategies"*. Systematically evaluating too many candidate QA strategies can require high effort. Thus, an effective pre-selection process is required to limit the evaluation effort for the involved stakeholders; *(b) Focus on domain-specific basic QA strategies*. Developing effective and efficient QA strategies typically focus on individual application domains. A set of domain-specific QA strategies and best-practices can help addressing individual domain needs and can increase scenario evaluation performance; (c) *Expert estimation & empirical evidence*. The estimation of expected benefits of individual QA strategies strongly depends on experience derived from prior strategy application and/or experts. Nevertheless, empirical data can be used as an additional source for (a) selecting most promising QA candidates and (b) estimating the expected benefits within a domain.

Success factors. Based on the success factors proposed by Stelzer *et al.* [20] we addressed these factors during QATAM design and application by involving relevant stakeholders, i.e., customers, management, and developers. In addition, we applied a stepwise and tailored improvement initiative by (a) providing a deeper understanding on the planned steps (and alternatives), (b) enabling a well-defined management of improvement projects, and (c) defining reasonable and reachable objectives and goals.

Future work. Based on the results of the evaluation future work will focus on a more detailed investigation and refinement of QATAM process steps. Additional follow-up effort is required to analyze changes and the long-term impact of these changes at the industry partner to get feedback on the strategy development and evaluation process.

References

[1] Basili, V., Caldiera, G., Rombach, D.: The Experience Factory. Encyclopedia of Software Engineering (1994)
[2] Beck, K.: Test Driven Development: By Example. Addison-Wesley Professional, Reading (2003) ISBN: 978-0-3211-4653-3
[3] Biffl, S., Denger, C., Elberzhager, F., Winkler, D.: Quality Assurance Tradeoff Analysis Method (QATAM): An Empirical Quality Assurance Planning and Evaluation Framework, Tech. Report, TU Vienna, IFS-QSE:0704 (2007), `http://qse.ifs.tuwien.ac.at/publication/IFS-QSE-0704.pdf`
[4] Biffl, S., Winkler, D., Höhn, R., Wetzel, H.: Software Process Improvement in Europe: Potential of the new V-Modell XT and Research Issues. Software Process Improvement and Practice 11(3), 229–238 (2006)
[5] Biffl, S.: Using inspection data for defect estimation. IEEE Software 17(6), 36–43 (2000)
[6] Deming, W.E.: Out of the Crisis. MIT Press, Cambridge (2000) ISBN: 978-0-2625-4115-2
[7] Denger, C., Elberzhager, F.: Basic Concepts to Define a Customized Quality Assurance Strategy, IESE Report No. 013.07/E (2007), `http://publica.fraunhofer.de/documents/N-55580.html`
[8] Denger, C., Elberzhager, F.: A Comprehensive Framework for Customizing Quality Assurance Techniques, IESE Report No. 118.06/E (2006), `http://publica.fraunhofer.de/documents/N-48500.html`
[9] Dyba, T.: An Instrument for Measuring the Key Factors of Success in Software Process Improvement. Empirical Software Engineering Journal 5(4), 357–390 (2000)
[10] Dyba, T.: An Empirical Investigation of the Key Factors for Success in Software Process Improvement. IEEE Trans. on Software Engineering 31(5), 410–424 (2005)
[11] Grünbacher, P.: Collaborative Requirements Negotiation with EasyWinWin. In: Proc. of 11th Int. Wsh. on Database and Expert Systems Applications, pp. 954–960 (2000)
[12] ISO 9001:2008: Quality Management Systems – Requirements (2008)
[13] ISO/IEC 15504-4:2004: Information Technology – Process Assessment – Part 4: Guidance on use for process improvement and process capability determination (2004)
[14] Kazman, R., Barbacci, M., Klein, M., Carriere, S.J., Woods, S.G.: Experiences with Performing Architecture Tradeoff Analysis. In: Proc. of ICSE, Los Angeles, pp. 54–63 (1999)
[15] Kläs, M., Elberzhager, F., van Lengen, R., Schulz, T., Goebbels, J.: A Framework for the Balanced Optimization of Quality Assurance Strategies Focusing on Small and Medium Enterprises. In: Proc. of Euromicro SEAA, Patras, Greece, pp. 335–342 (2009)
[16] Niazi, M., Wilson, D., Zowghi, D.: A Maturity Model for the Implementation of Software process improvement: an empirical study. Software and Systems J. 74(2), 155–172 (2005)
[17] Rainer, A., Hall, T.: Key success factors for implementing software process improvement: a maturity-based analysis. Software and Systems Journal 62(2), 71–84 (2002)
[18] Rus, I., Halling, M., Biffl, S.: Supporting Decision-Making in Software Engineering with Process Simulation and Empirical Studies. IJSEKE 13(5), 531–545 (2003)
[19] Schwaber, K.: Agile Project Management with Scrum. Microsoft Press, Redmond (2004) ISBN: 978-0-7356-1993-7
[20] Stelzer, D., Mellis, W.: Success Factors of Organizational Change in Software Process Improvement. Software Process Improvement and Practice 4(4), 227–250 (1999)