SADIO Electronic Journal of Informatics and Operations Research

http://www.sadio.org.ar

vol. 10, no. 1, pp. 68-77 (2011)

Lightweight framework for quality assurance in SMEs

Ariel Schapiro[1] Nicolás Paez[2]

[1] Southworks SRL
e-mail: ariel.schapiro@southworks.net

[2] Facultad de Ingeniería, UBA
e-mail: nicopaez@computer.org

Abstract

Based on their business needs, many software Small and Medium Enterprises (SMEs) differentiate through the high quality of their deliverables, compliancy with standards and alignment with engineering best practices. This paper explains how an SME in that context, successfully used a lightweight framework based on premises like self-assessment, tailoring, automation and positive peer pressure that assured a high level of service quality while removing the implied costs that would be derived from implementing a Quality Assurance (QA) department. The framework aligned with the company's agile processes allowing teams, through the implementation of short iterations, to assess their compliancy with a tailored quality baseline and make reviews with the help of cross-teams Quality Reviewers. The result was a low cost framework that helped to grow factors like overall process quality while increasing quality perceived from the customer.

Keywords: quality measurement, quality engineering, quality assurance, Small and Medium Enterprise (SME), agile.

1 Introduction

We work at an Argentinean SME that provides architects, developers, project leaders and product specialists to leader companies in the US, Europe and Latin America. A considerable part of our services includes the development of reference applications and solutions that implement emerging technologies, published as examples to follow in terms of quality standards and best practices. These best practices range from architecture principles and design patterns to technical writing rules and user experience recommendations.
Having participated of the development of guidelines and technical reference materials made us aware, from the beginning, of the important role of meeting high quality standards. On the other hand, technical limitations of beta products and rapid changes in scope were part of the environment imposed by working with those emerging technologies. The need to enable flexibility to determine priorities along the way, helped us to quickly adopt agile methodologies, implementing a process where changes are applied in weekly iterations.
A more global background may include software services that turn into commodities and the increasing reliability of outsourcing. Those factors also contributed to the decision of differentiating through a high quality products and services.
Our company certified ISO 9001 a couple of years after implementing the aforementioned process. At the same time, we spiked with some initiatives related to CMMI, Six Sigma and People CMM. Since many of these models may require a higher deployment cost that would include initiatives like growing our small-scale process area, those initiatives remained unofficial.
Differently from Crosby's suggestion about "Free quality" concept [1] and aligned with Scott Johnson [2], we realized that in our case the implementation of a quality management system implies an up-front cost that ultimately affects organizational projects. In this context of having implemented an agile process and being certified by ISO standards, in time we incorporated several indicators to higher the quality bar of our product and service quality.
Throughout our history, we were always imposed by the challenge of adopting practices to ensure the quality needed by our business but at the same time trying to minimize the cost of implementation at the organization and project levels.
The next sections will showcase the process used to guarantee a high level of service quality and at the same time lower the implied costs; its characteristics and the results of implementing it in our organization.

2 Premises of our quality assurance process

Our quest on minimizing the costs for adopting practices to ensure and improve the quality of our products, has lead us to four premises explained below.

2.1 One size does not fit all

Our organization faces different types of projects, each one with distinctive properties and quality indicators. For instance, building a line of business application might defer from writing a guide for performance testing. Evaluating this array of project types with the same set of quality indicators might carry efforts not directly tied to value-added situations on all projects.
Therefore, tailoring the baseline of quality indicators by project type seems to be a good approach for getting the most of such measurements.

2.2 Self-assessment

This practice is motivated by several factors. We encourage our engineers to be a key role in terms of quality assurance, as opposed to scenarios where quality is injected by third parties to the product. This is aligned with the Lean principle that recommends building quality in [3]. At the same time self-assessment is part of the self-organized teams mindset proposed by agile methods [4].
In our context of small teams, by taking the quality into their hands, engineers tend to become accountable for the product quality, instead of depending on a relatively more costly separate QA department.

2.3 Automation

In the context of agile methods, automation is commonly used to perform testing-related tasks [5]. Following this mindset and the spirit of committing resources to only value-added outcomes, we try to follow manual steps when they represent creative or intellectual tasks only.

By automating tasks related to our quality assurance process and through the usage of handy applications, we tend to lower our costs and avoid errors that may arise from repetitive steps.

While some automation tools were developed by people in the context of internal projects, most of them arose from the experience of teams working in projects for our customers: they wanted to reduce the amount of errors from repetitive manual tasks.

2.4 Positive peer pressure

Each project counts with an owner who is accountable for the quality of its product and processes, but results of measurements around his area of responsibility are not delivered on a private manner, not even only to the team he advocates for. We realized that if we publish all the projects measurements and list the results including the name of each project or responsible, we helped them to see not only their results, but their peers too.

This demonstrated to become an important factor in the search for improvement of every project, as each project member can easily contrast their results against the rest of the organizations'. In a situation where a project does not meet with a certain indicator, it can easily become aware of which other project does comply with the indicator and then ask for specific help.

We became aware of the fact that this method helped on the creation of informal channels and like Jon R. Katzenbach explains [6], we believe that this positive peer pressure made more natural for team members to connect with each other in order to improve their service quality

3 Lightweight Quality Assurance Process

Our particular scenario, related with the experience of building high quality software and teaming with our customers, encourages us to think of quality beyond the user-product relationship and into a concept composed by several dimensions or points of view. We group those into three dimensions: product, process and project.

When talking about software in the dimension of product quality, we often consider conformance to requirement, defects count or user experience. Other internal product properties like code analysis, source analysis and code coverage are of value, especially in our setting where part of our deliverables are reference applications for developers or engineers.

Process quality is mentioned in CMMi-like approaches that consider the quality of the product to be determined by the process followed when building it. Process Quality in our case includes concepts related with the service we provide, such as level of communication held with the customer, planning or risk management practices, etc.

Project quality dimension cross-cuts the later two and complements them. It includes the view from the customer or internal stakeholders regarding support level, ROI measurements, customer satisfaction, etc

Fig. 1. Product, Process and Project points of view interacting from different but complementary quality parameters.

Even in this proposed scenario, where the customer experience is highly valued, it may happen that at first, a customer is attracted by the product quality features: the external view, the output of the organization's processes. After the initial contact, as the customer is part of a project, it may well happen that the customer is involved in the processes that lead to that product and therefore expects to receive a quality experience that surpass the software deliverable itself.
Each component may provide a different value: while a poor quality product can cause a customer to be dissatisfied, the process and the project dimensions provide the experience that the customer can perceive as a differentiator.

We believe this approach can be applied in SMEs as it is described in this work. For larger companies, the approach would require some modifications due to common policies applied by large enterprises. At the same time, large companies tend to conform bigger teams where communications does not flow in the same way and the development team has little interaction with the customer.

3.1 Quality baseline and tailoring

For each of the previously mentioned quality dimensions we have identified a set of relevant concerns to be considered for defining the overall quality level of the dimension.

For example if we look at the product quality dimension we may find concerns like code coverage, code analysis and source analysis. If we consider the process quality dimension, concerns like risk management, planning and customer involvement may appear. Finally, the project quality dimension might include concerns like profitability, business alignment and customer satisfaction.

Once we defined a set of quality concerns with those generally relevant to our organization, we considered the premise of not all of them fitting for all projects and hence identified the different types of projects we usually work with. The result is a quality baseline that basically defines and explains the level of relevance of each concern for each project type.
This way we have created a quality baseline for each quality dimension, including a matrix like the ones shown in the following sections and guidelines provided to ensure each concern to be understood and tackled appropriately

3.1.1 Product quality

There are plenty of definitions of quality but most of them are related to this quality dimension. According to Juran[8] the 2 most important quality definitions are:

- Product features, in the eyes of customers: the better the product features the higher the quality.
- Lack of deficiencies: the fewer deficiencies the better the quality.

While the first of these definitions includes concerns like user experience, the second is in direct relation with metrics like defects count. We agree with both definitions but our view of the product quality is broader, paying also attention to certain internal properties like code analysis, source analysis and code coverage.

We find these properties very important for the software evolution. If we consider the whole software lifetime, the maintenance cost is often much bigger than the construction cost and is here where all these internal properties can make a huge difference. Let's consider the implementation of a new feature once the software has been released. This new feature will require some modifications to the original code base; is it possible to ensure that these modifications do not inject any new defects in the product? If every time a modification is introduced into the system, a set of automated tests are run with a high percentage of code coverage, then the possibility of injecting new defects is lower.

Concerns	Training kit	Sample application	Reference application	Proof of concept
Security	O	M	M	O
Web performance	O	R	M	O
Web standards	R	M	M	O
Code analysis	R	M	M	R
Source analysis	R	M	M	R
Unit tests	O	R	M	R
Code coverage	O	R	M	R
Documentation	M	M	M	O
Install experience	M	M	M	R

Table 1. Product quality baseline per project type. M=mandatory, R=recommended, O=optional

Let's review an example to understand the meaning of the table above. The web performance concern is mandatory for a Reference Application project, meaning that the associated guidelines must be followed. At the same time, the same concern is optional for a project where a Training Kit is released, meaning that the associated guideline can be ignored. Finally, a project that delivers a Sample Application includes the web performance concern as Recommended: the concern should be considered but the implementation of the guidelines for that project will depend on its specifics. Hence, the team should make a decision and document it.

3.1.2 Process quality

The process quality dimension is focused at the compliancy of soft practices that compose our standardized processes. In this case in particular, most of them are mandatory across project types, as they are part of a Quality Management System aligned wih ISO standards. The fact that most of those concerns are mandatory relies on the reality that they are generally born out of the methodology principles of the organization that, in this case is aligned with agile principles. That concept may explain the importance given to concerns like *frequent status updates* and *frequent releases*.

Other concerns are more related with classical project management interests such an appropriate *risk analysis,* the *documentation of design decisions* and counting with an updated *release plan*.
There are other items that are not mandatory across project types, below the dotted line in the next table. That area also belongs to the concerns that are likely to be added to the framework, without necessarily requiring all teams to comply.

Concerns	Training kit	Sample application	Reference application	Proof of concept
Risk Analysis	M	M	M	M
Documentation of design decisions	M	M	M	M
Frequent status updates to team and	M	M	M	M

customers				
Release plan	M	M	M	M
Project kick-off communication	M	M	M	M
Project closure communication	M	M	M	M
Frequent releases	M	M	M	M
	M	M	M	M
Team retrospective meetings	M	M	M	M
Press communications	R	R	R	O
Reuse of specific components	R	R	R	O

Table 2. Process quality baseline per project type. M=mandatory, R=recommended, O=optional

3.1.3 Project quality

The quality dimension designated with the "project" alias, is a group of concerns that correspond to the customer or internal stakeholders, aligned with measurements of tools like balanced scorecards. The project quality may have a broader perspective, compared with the product and process viewpoints. It may be related with aspects like financials, business and company growth. That may be a reason to assume that those global parameters are present in all projects, but like we will see in the next section, each team has the choice to modify this at the beginning of the project during the tailoring.

Concerns	Training kit	Sample application	Reference application	Proof of concept
Customer Satisfaction	M	M	M	M
ROI	M	M	M	M
Strategic opportunities	M	M	M	M
Employee satisfaction	M	M	M	M
Knowledge additions	M	M	M	M

Table 3. Project quality baseline per project type. M=mandatory, R=recommended, O=optional

3.2 The process itself

The process of implementing the Lightweight Quality Assurance Process consists of five steps that each project will follow to measure and follow their quality concerns. Step 1 appear at the beginning of the project life cycle, step 5 at the end and steps 2,3,4 are repeated with a frequency defined by each project depending mainly on its length but most of our projects use cycles of 2 weeks.

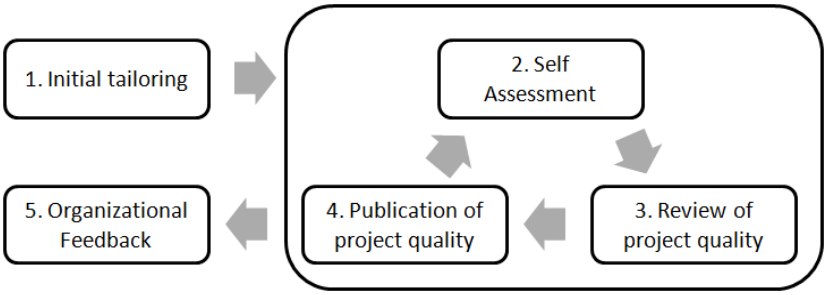

Fig. 2. The process includes 5 steps. Steps 2,3 and 4 are repeated several times along the project.

In the context of this process we consider the following roles:

- Project team member: each person that works in the project most of his time (assignment > 80%).
- Project Quality owner: a team member responsible for complying with the committed quality levels for that project. This does not mean that this person has to perform all the quality related tasks, it just means that he will have to pay a special attention
- Quality Reviewer: not part of the team, with enough knowledge and experience on the quality concerns that apply to the project. The Quality Reviewer role does not require a full time availability; most of the time this person works as a team member on a different project.

As with the Quality Reviewers, having some team members committed with their projects but also contributing on other project's quality, helps them become more accountable on the quality advises they give. Applying this model is also an important factor on not needing a separate and full time committed department for quality engineering.
The following sections explain the steps described in the image above

3.2.1 Initial tailoring

When the project kicks off, team members together with a Quality Reviewer tailor the quality baseline explained in point 3.1 to their project context. Most likely they'll discuss which of the recommended or optional concerns will be applied and review the guidelines for all the quality concerns involved in the project.

3.2.2 Self-assessment

Every 2 weeks, the Project Quality Owner, following the quality guidelines reviewed in the previous step, performs a self-assessment to determine the current quality level of the project, product and process dimensions. This activity in most cases consists of using the guidelines provided by the quality baseline to check for each of the quality concerns compliancy. Some concerns might include measurements to be taken automatically by a certain tool, like code analysis

3.2.3 Review of Project quality

Afterwards, team members together with the Quality Reviewer, analyze the quality measurements of the project delivered in the self-assessment. They detect issues, plan for corrective actions and adjust the project quality bar.

3.2.4 Publication of Project quality

Once reviewed, the project quality status, along with the issues and plans for corrective actions are shared with the customer. This could be part of sprint review meetings that are part of the agile process each team also follows.
Optionally, this can also be shared with the rest of the organization, in order to benefit other teams working on similar quality concerns

3.2.5 Organizational feedback

When the project is completed, lessons learned are shared with the rest of the organization, through a presentation that also includes tailoring details of the quality baseline and the progress of their quality concerns.

Additionally, based on the project experience, the team proposes improvements to the quality baseline, pushing for continuous improvement of the organization services quality.

3.3 The quality guidelines

For each quality concern, organizational guidelines are provided to guide the teams with the implementation of the concerns and also with their assessment. These guidelines are based on industry best practices and were originally collected by projects teams during the execution of previous projects. The update of these guidelines and the incorporation of new ones, is done with the same strategy. This way the cost of generating/updating these guidelines is absorbed by the projects without representing any additional cost for the organization.

4 Results and Conclusions

As we suggest in this area, the cost of implementing this framework proved to be relatively low in comparison with the added value represented by the improvement of the global indicators linked to the initiative.
Principles like the self-assessment helped on keeping a low cost for running the framework while factors like overall process quality and quality perceived from the customer increased.
In the next sections, we'll present a detail of the costs derived from implementing the framework and the benefits obtained.

4.1 Costs

As we described in the introduction, one of the main goals of having this framework running was to minimize its cost. The design arose from everyday experience as part of initiatives in isolated projects. Over time, those initiatives proved to be valuable to the organization and therefore we decided to formalize them.
On the other hand, once running, this framework implies, although minimal, certain costs derived from either the quality check process or the enhancement of the framework itself.

We can estimate the efforts related to those costs as the following:

- Framework design: one-off effort of 160 man-hours from a group of senior collaborators.
- Framework implementation (per project effort):
    1. Initial tailoring: 2 hours of every team member.
    2. Self-assessment: every 2 weeks, 2 hours of the quality owner of the project.
    3. Review of Project Quality: every 2 weeks, 2 hours of the quality reviewer with all of the team members.
    4. Publication of project quality: this step is highly automated, thus we opt to minimize it.
    5. Organizational Feedback: once at project closure. 2 hours of all team members.

Assuming that a project headcount remains stable, on a project basis, we can summarize these efforts as a weekly investment in hours for each team member based on the following equation $I(w,n)$, where w represents the duration of the project in weeks and n the amount of team members

$$I(w,n) = (4 / w) + [(3 + n) / (2*n)]$$

For example, one project with 5 team members and duration of 6 months (~24 weeks) would impose an effort of less than an hour for each team member per week.

4.2 Benefits

The first consequence from implementing this framework is the fact that we started measuring important quality aspects of our projects, products and processes. Aside from quality expert James Harrington's quote

"…If you can't control it, you can't improve it." [7], the fact of measuring themselves started adding value by letting teams know where their position in terms of quality was and how to reach the next step.

One of the conclusions we made is that when measuring concerns related with process, the overall process quality of the project is expected to grow during the lifetime of the project: teams build assets and acquire practices that help them increment their process quality rate. Based on these presumptions, we analyzed more than 350 process measurements in a period of 10 months and specifically found that:

- When kicking off, projects in average comply with 55% of the quality concerns and during the first 6 weeks, 68% (1 sigma) of projects overall process quality rate is either 15% above or below that average line (light blue, surrounding the average in the chart).
- **In average a project grows in process quality terms 3% per week during its first 6 weeks; then 0.5% for the following 10 weeks.**

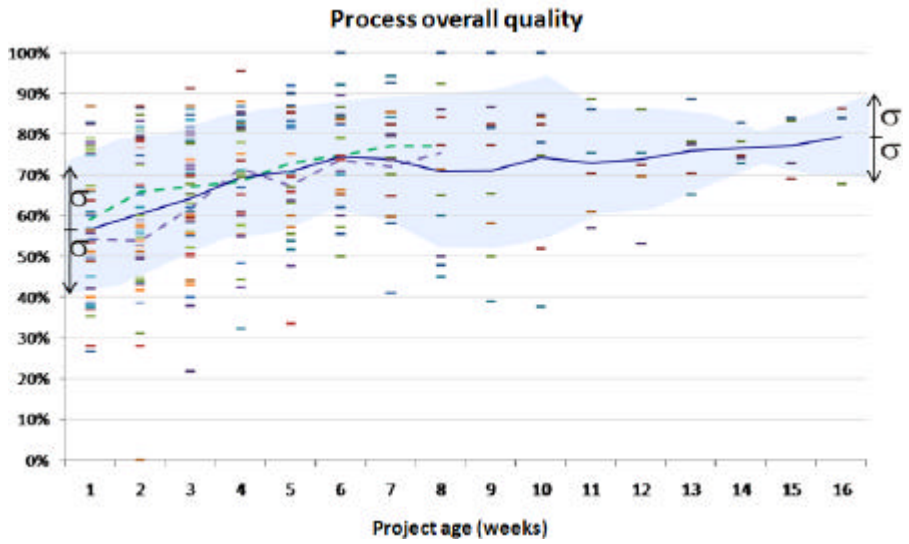

Fig. 3. Overall process quality evolution for projects based on their age in weeks.

While the enhancement of quality based on constant measurement improved the overall quality as we just proposed, we understand that by communicating this quality level to the customer iteratively could have also impacted on the perception from the customer in terms of the quality of our service.

Our customer satisfaction surveys showed an improvement in perceived quality areas after the first year of having gradually implemented the framework. In particular, we found an increase in the perceived quality of our service and deliverables of 6% in average and we also noted a decrease in the amount of complaints from our customers: from 53% of them making any kind of complaints on a rarely basis, down to 29%. It is worth to highlight that the rest of the customers never reported to make any kind of complaints, ever.

References

1. Crosby, P.B: Quality Is Free. Penguin Books, New York (1980)

2. Johnson, S.: Quality is not free, http://c2.com/cgi-bin/wiki?QualityIsNotFree

3. Poppendieck, M., Poppendieck, T.: Implementing Lean Software Development: From concept to cash. Addison-Wesley Professional, Massachusetts (2006)

4. Schwaber K., On Self-organizing teams. http://www.controlchaos.com/storage/scrum-articles/selforg.pdf

5. Crispin L., Gregory J., Agile Testing: A Practical Guide for Testers and Agile Teams. Addison-Wesley Professional, Massachusetts (2009)

6.Harvard Business Review. Positive Peer Pressure: A Powerful Ally to Change, http://blogs.hbr.org/cs/2010/04/positive_peer_pressure_a_power.html

7. Spitzer, D.R.: Transforming Performance Measurement: Rethinking the Way We Measure and Drive Organizational Success. AMACOM, New York (2007)

8. Juran, J. M: Juran on Quality by Design, Free Press