



Continuous Improvement in Software Development

Lori Holmes, Director

Total Quality Management (TQM) is a buzzword heard today throughout all industries. It involves many concepts and ideas. This article will address the TQM concept, continuous improvement, as it applies to software development.

Typically in software development the focus is on problem solving. In addition, continuous improvement involves examining processes to proactively determine improvement opportunities. Both problems and opportunities can be addressed using the same methodology. The steps involved are similar to what software professionals typically follow on a day-to-day basis. TQM provides the framework to formalize these steps. The TQM framework for continuous improvement does not need to be complicated.

Often organizations implementing TQM make the mistake of requiring continuous improvement to be formal, time consuming and to involve many people. Part of making continuous improvement continuous is to make it easy for employees to make changes as they deem necessary. A process should always be followed, but complexity and duration is unique to each situation. For example, improving estimating or productivity may involve several people and take time to break down the issues. On the other hand, changing communication processes between departments can be handled quickly with minimal involvement. Each contributor must determine how involved the process should be based on his or her knowledge of the issue and the organization.

The formal continuous improvement methodologies that are most successful involve the following steps: 1. Describe the Issue; 2. Determine the Cause; 3. Resolve the Issue; and 4. Follow-Up. The following addresses each step in more detail.

DESCRIBE THE ISSUE

This step involves two parts: identifying the issue and describing the issue. A variety of sources are available to identify software improvement opportunities. These ideas can come from trouble reports, customer complaints or employee ideas. A more formal means of identifying opportunities is to utilize information gathered throughout the software life cycle. This information can come from techniques such as defect analysis, post project reviews or from reviewing project attribute data. Software professionals should use all of these methods to continually look for improvement opportunities.

Once an issue has been identified it is important to fully describe it and quantify the consequences of making changes. The consequences can be the cost of not fixing a problem or the cost savings resulting from the improvement. These cost savings can be due to reduced rework or increased quality and productivity.

Collecting cost information aids in justifying and quantifying changes from a business perspective.

DETERMINE THE CAUSE

It is important to get to the root of the problem or opportunity so that it will be resolved permanently. Often symptoms are addressed instead of causes, so problems linger and improvements are not completely effective.

Some of the techniques to determine these root causes are: process flow analysis; requirements reviews; cause/effect or fishbone diagramming; and measurement. Often, existing data can show where process changes will be most effective. These techniques are useful when approaching large involved changes. For smaller concerns, brainstorming is an effective technique.

RESOLVE THE ISSUE

Resolutions may be easy to determine or they may involve brainstorming or research. It is imperative to understand individual authority and responsibility to determine what level of commitment and involvement from others is necessary. In addition, impacts on time, cost, customers, suppliers and schedules should be analyzed to select the appropriate action.

The implementation of the resolution should be planned to avoid problems. This can be a list of a few items for simple changes or an involved project plan for larger changes. It involves specifying activities, responsibilities and time frames for making process changes. This not only helps with implementation, but ensures resources are available when following up on the improvement.

FOLLOW-UP

This step is important and often overlooked. It is necessary to make sure changes have been implemented correctly and that the desired outcome was achieved.

Follow-up enables organizations to show successes and demonstrate the impact of improvements. It involves measuring the impact against the initial

objectives. More specifically, results are measured to see if costs have been reduced or avoided and whether defects have decreased or productivity has increased. Any savings large or small is worth noting. Often, past success enables future opportunities to be addressed.

SUMMARY

The 4 steps described for continuous improvement are followed by software professionals every day when dealing with problems such as abends and incident reports. In one aspect this is an advantage for software professionals implementing continuous improvement. However, to fully benefit from this concept it needs to be applied to all processes, not just problems. The focus needs to be on how things are done and not just on fixing problems.

It also must be addressed on a continuous basis. Improving processes should be part of what is done every day. As industries, customers and technologies grow and change, organizations need to move ahead as well. Often what has worked in the past will not work in the future. In addition, all companies are looking to "do more with less." Individuals must be willing to look for ways to resolve problems and improve processes so their jobs will be efficient, effective and meet company needs.

About the author

Lori Holmes, a Director with Q/P Management Group, Inc., wrote this article. Ms. Holmes specializes in software measurement, process improvement, and quality assurance. Her areas of expertise include function point analysis, software project estimation, establishing measurement programs, software quality assurance, and managing customer satisfaction.