Software Quality Management – At a Glance (SEI View)

Muhammad Mahbub Hussain
Deputy Team Leader (SQA)
IMS Health Inc. / M&H Informatics (BD) Ltd
September, 2006

Arguably though (with minor negations), Software Quality Management (SQA) is the most important activity of a software development life cycle. Again, Testing is the one of the most important Quality Management activities, and these two in combination, produces a group of activities which is aimed at the evaluation of the capability, maturity, and conformance to user requirements of any software to be produced or developed.

COST OF QUALITY

WE CAN DO IT RIGHT, OR WE CAN DO IT OVER AND OVER AGAIN.

If we can stress quality in all engineering activities during the lifecycle of a product or service, it reduces the amount of rework that we must do, or perhaps, we often do. If we can reduce the amount of rework, it would result in lower costs, and more importantly, improved reputation due to improved time-to-market. Let us take a closer look at different aspects of the *Cost of Quality* in the following paragraphs.

Quality costs may be divided into cost associated with –

- Prevention
- Appraisal, and
- Failure.

Prevention costs include –

- Quality Planning
- Formal Technical Reviews (FTRs)
- Test Equipments, and
- Training.

Appraisal costs include –

- In-process Inspection
- Inter Process Inspection
- Equipment Calibration and Maintenance, and
- Testing.

Failure costs are those that would disappear if no defects appeared before shipping a product to customer. Failure costs may be sub-divided into internal and external failure costs.

Internal Failure costs include –

- Rework,
- Repair, and
- Failure Mode Analysis of defects found prior to shipment.

External Failure costs include –

- Compliant resolution,
- Product return and replacement,
- Help line support, and
- Warranty work for defect found after shipment.

SQA ACTIVITIES

The Software Engineering Institute (SEI) recommends a set of SQA activities, and they are –

1. Prepare an SQA Plan for the project.
2. Participates in the development of the project's software process description.
3. Reviews software engineering activities to verify compliance with the defined software process.
4. Audits designated software work products to verify compliance with those defined as part of the software process.
5. Ensures that deviations in the software work and work products are documented and handled according to a documented procedure.
6. Records any noncompliance and reports to the senior management.

SOFTWARE REVIEWS (WALKTHROUGH OR FORMAL TECHNICAL REVIEW)

Objectives of Formal Technical Reviews (FTRs) are –

- Cost Impact of Software Defects
- Quality factors define and analysis
- Defect amplification and removal

Minimum set of focus areas of an FTR is –

- Review the product, not the producer.
- Set an agenda and maintain it.
- Limit debate and rebuttal.
- Enunciate problem area, but do not attempt to solve every problem noted.
- Take written notes.
- Limit the number of participants and insist upon advance preparation.
- Develop a checklist for each product that is likely to be produced.
- Allocate resources and schedule time for FTRs.
- Conduct meaningful training for all reviewers, and
- Review all earlier reviews.

Statistical Software Quality Assurance

For software, statistical quality assurance implies the following steps:

1. Information about software defects is collected and categorized.
2. An attempt is made to trace each defect to its underlying cause (e.g. non-conformance to specifications, design error, violation of standards, poor communication with customers etc.)
3. Using the Pareto Principle (80 percent of the defects can be traced to 20 percent of all possible causes), isolate the 20 percent (the "vital few").
4. Once the vital few causes have been identified, move to correct the problems that have caused the problems.

Generic Template of Statistical SQA

All defects can be tracked to one or more of the following causes:

1. Incomplete or erroneous specification (IES).
2. Misinterpretation of customer communication (MCC).
3. Intentional deviations from specification (IDS).
4. Violation of programming standards (VPS).
5. Error in data representation (EDR).
6. Inconsistent component interface (ICI).
7. Error in design logic (EDL).
8. Incomplete or erroneous testing (IET).
9. Inaccurate or incomplete documentation (IID).
10. Error in programming language translation of design (PLT).
11. Ambiguous or inconsistent human/computer interface (HCI).
12. Miscellaneous (MIS).

Sample of data collection table for statistical SQA

	Total		**Serious**		**Moderate**		**Minor**	
Error	**No.**	**%**	**No.**	**%**	**No.**	**%**	**No.**	**%**
IES								
MCC								
IDS								
VPS								
EDR								
ICI								
ELD								
IET								
IID								
PLT								
HCI								
MIS								
Totals								

Statistical SQA – a Six-Sigma (for SE) view

Six-Sigma is the most widely used strategy for statistical quality assurance in industry today. The six sigma methodology defines three core steps:

1. *Define* customer requirements, deliverables and project goals via well-defined methods of customer communication.

2. *Measure* the existing process and its output to determine current quality performance (collect defect metrics data).

3. *Analyze* defect metrics and determine the vital few causes.

If an existing software process is in place, but improvement is required, six-sigma suggests two additional steps:

4. *Improve* the process by eliminating the root causes of defects.

5. *Control* the process to ensure future work does not reintroduced the causes of defects.

(Note: This process is known as DMAIC process well)