ISSN: 2248-9622 www.ijera.com
Vol. 2, Issue 1,Jan-Feb 2012, pp.552-557

Improving Quality of Software through Formal Inspection

T.Rajani Devi

ABSTRACT
The software inspection process was created for the dual purpose of improving software quality and increasing programmer's productivity. Formal inspection is a better method than technical walkthroughs in the software life cycle process. benefits gained in the development of defect-free software by utilizing formal inspection Formal inspections may be applied to any product or partial product of the software development process, including requirements, design, and code. The software formal inspections are in-process technical reviews of a product of the software life cycle, which aims to detect and eliminate defects in the early stages of each products development. In addition, the evaluation from the formal inspections can be immediately fed back to the author with improvements in the quality of future products.

Keywords- Appraisals, Fagan Inspection, Formal inspection, Milestones, Quality assurance, Walkthroughs.

INTRODUCTION

The software inspections were first introduced by Michael E. Fagan in 1970s, when he was a software development manager at IBM. The inspection process for software development was also developed as it was practiced by IBM at that time. To distinguish the software inspection from general inspection, the software inspection should be called the in-process inspection. For simplicity, we frequently use the term "inspection" as synonymous with "in-process inspection". Software inspections allow software development teams to find defects earlier and cheaper, thus reducing rework costs. In addition, there are other benefits more difficult to quantify. Software inspections aid in project management; and they provide more definite and more dependable milestones than less formal review processes. Software inspections can also promote closer teamwork, provide on-the-job training, and support the transfer of skills from more experienced team members to others. There are some costs to starting up inspections, specifically in training management, moderators, and inspectors. The conduct of trained moderators and the attitude of management are key to the acceptance of inspections by engineers. Inspection results should not be used for personnel performance appraisals. An inspection is one of the most common sorts of review practices found in software projects. The goal of the inspection is for all of the inspectors to reach consensus on a work product and approve it for use in the project. Commonly inspected work products include software requirements specifications and test plans. In an inspection, a work product is selected for review and a team is gathered for an inspection meeting to review the work product. A moderator is chosen to moderate the meeting. Each inspector prepares for the meeting by reading the work product and noting each defect.

The goal of the inspection is to identify defects.

In an inspection, a defect is any part of the work product that will keep an inspector from approving it. For example, if the team is inspecting a software requirements specification, each defect will be text in the document which an inspector disagrees with. There are various names for the same thing. Some call it software inspection, which also could extend to the design and its documentation; some call it code inspection which relates more to the source code. A third name would be Fagan Inspection, called after the person who invented this quality assurance and testing method. Code inspections are a highly efficient test method which cannot be substituted by any other test methods. It is time consuming but according to statistics it will find up to 90% of the contained errors, if done properly. However it all depends on the methods and checks applied and on the diligence of the inspectors. It must not be confused with the so called "code review" or "walk through" which is usually done in a single meeting lasting for a couple of hours. A proper code inspection may take several days and needs the help of tools to browse the symbols in order to find the places where they are used. The code review can be used in addition to e.g. to generated acceptance of a software package by the integrators, but it must not be a substitute for a proper inspection. Proper inspections can be applied for almost all work products in the software life cycle. At the first glance they may look very time consuming. But statistical evaluations have shown that over the whole life cycle of the software development they even save resources and thus money and improve the quality of the product.

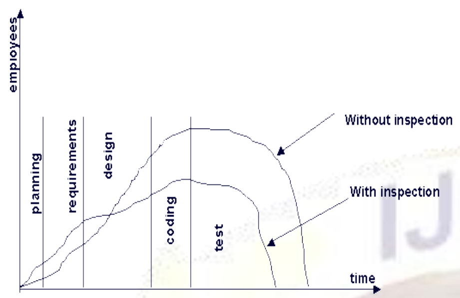

Fig. (1) Source of the diagram: Michael Fagan

Most people are not aware that the manual static testing methods i.e. inspections, reviews and walkthroughs are defined in the "IEEE Standard for Software Reviews". This is the IEEE 1028-1997 standard. I want to give a short overview on the main definitions in this standard; however I will not discuss the "Management Review" which is in the widest sense a check of a project's performance and the related documents. I will also omit a discussion of the Audits described in the standard, which a more related to having external checks on work products and processes. I will focus on the review techniques for technical work products as they are typically used within a company. I also want to point out the problems involved with these methods and make an attempt to present some solutions for these problems.

1. The process

The inspection process was developed by Michael Fagan in the mid-1970s and it has later been extended and modified. The process should have entry criteria that determine if the inspection process is ready to begin. This prevents unfinished work products from entering the inspection process.

The stages in the inspections process are: Planning, Overview meeting, Preparation, Inspection meeting, Rework and Follow-up. The Preparation, Inspection meeting and Rework stages might be iterated.

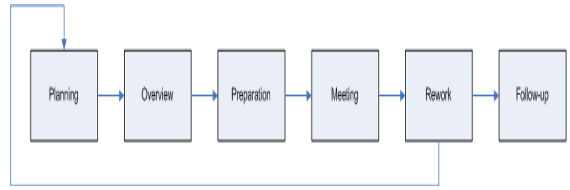

Fig .(2)

A formal inspection consists of several activities:

1.1 Planning- The period of time used to determine whether the product to be inspected meets the entry criteria, set the inspection schedule, plan the inspection itself, select a team of inspectors and assign respective roles, and prepare and distribute the inspection materials. This is when the moderator decides whether an overview will be necessary, as well.

1.2 Overview- An optional stage in the inspection process. The overview provides the inspection team with background information for the inspection. This stage may not be necessary if the team is already intimately familiar with the work product being inspected.

1.3 Preparation- A Key stage during which each member of the inspection team individually prepares for the inspection. It is crucial that individual inspectors be given adequate time to prepare, otherwise the inspection process will not be efficient and may well fail to identify defects that could otherwise be discovered. Each team member prepares for the inspection by reviewing and finding potential defects in the product being inspected before the inspection meeting. Potential defects are then discussed during the inspection meeting as a group.

1.4 Inspection Meeting- Meeting where team members as a group review the product to discover, categorize and record defects. Defects are not resolved during this meeting.

1.5 Third Hour—Literally, a third hour to the inspection meeting (the formal inspection meeting is limited to two hours). This is an optional additional time that can be used to discuss, possibly solve or further investigate defects that have already been discovered during the Inspection Meeting.

1.6 Rework—Stage when the author makes changes to the work product and corrects defects according to the action plans from the inspection meeting.

1.7 Follow-up—a short meeting between the author and the inspection moderator used to determine if the defects found during the inspection have been corrected and to ensure that no additional defects have been introduced.

To qualify as a true inspection, the activity follows a specified process and the participants play well-defined roles. An inspection team consists of 3-8 members.

2. A Formal inspection team includes these roles

2.1 Moderator

Responsible for administrative tasks, schedules meetings, controls the meetings, reports inspection results, and follows up on rework issues. Moderators should be trained in how to conduct inspections, including how to keep participants with strong technical skills but low social skills from killing each other.

2.2 Author

Created or maintains the work product being inspected. The author may answer questions asked about the product during the inspection, and he also looks for defects. The author cannot serve as moderator, reader, or recorder.

2.3 Reader

Describes the sections of the work product to the team as they proceed through the inspection. The reader may paraphrase what is happening in the product, such as describing what a section of code is supposed to do, but he does not usually read the product verbatim.

2.4 Recorder

Documents issues, decisions, and recommendations. Records inspection data for process analysis. The moderator might perform this role in a small inspection team.

2.5 Inspector

Attempts to find errors in the product. All participants actually are acting as inspectors, in addition to any other responsibilities. Good people to invite as inspectors include: the person who created the predecessor specification for the work product being inspected (e.g., the designer for a code inspection); those responsible for implementing, testing, or maintaining the product; a quality assurance representative to act as standards enforcer; other project members; and someone who is not involved in the project at all but who has the skill set and defect-detection abilities to be able to contribute usefully to inspecting any work product of this type. We also require that our key customer representatives participate in requirements specification inspections. Familiarizes himself/herself with the artifact to be inspected and identifies issues with it

3. Participation of inspectors

3.1.Planning
Roles:
-Moderator
-Author

3.2.overview
Roles:
-Moderator
-Author
- Inspectors

3.3.Preparation
Roles:
- All inspectors

3.4. Inspection Meeting
Roles:
- Moderator
- Author
- Reader
- Recorder
- Inspectors

3.5. Discussion
Roles:
- All inspectors

3.6. Rework
Roles:
- Author

3.7. Follow Up
Roles:
- Moderator
- Author

4. RELATIONSHIPS TO OTHER PRACTICES

The Figure below represents ahigh-level process architecture for the subject practice, depicting relationships among this practice and the nature of the influences on the practice (describing how other practices might relate to this practice). These relationship statements are based on definitions of specific “best practices” found in the literature and the
notion that the successful implementation of practices may “influence” (or are influenced by) the ability to successfully implement other practices. A brief description of these influences is included below.

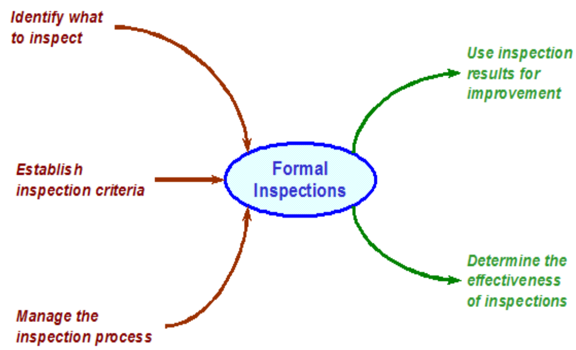

Fig .(3)
Process Architecture for the "Formal Inspections" Gold Practice

5. INPUTS TO THE PRACTICE

5.1 Identify what to inspect:

Several practices influence the Formal Inspection (FI) practice by providing artifacts for inspection. Leveraging COTS/NDI typically includes an analysis process. Inspections can be performed on analysis results (or artifacts of the analysis process) in preparation for technical decision making regarding COTS selection or implementation. Applying FI to Reuse Components can identify issues and problems relating to reuse of the components in the new environment thus mitigating some of the risks. Agreement on Interfaces addresses identifying all the interfaces and understanding their functions. The artifacts used to represent the interfaces should be inspected for accuracy and completeness as part of the agreement process. The Architecture-First Approach is an iterative approach that produces technical artifacts (architectural diagrams from a variety of perspectives, or demos), which are reviewed by key stakeholders with each iteration resulting in refinement until the architecture becomes stable. Embedding a formal inspection practice into the review cycle will likely find flaws in the architecture. Feedback from the results of inspections can then be used to improve the architecture. The Manage Requirements practice influences formal inspections in two ways:

5.1.1 Requirements documents are subjected to formal inspections to improve the quality and completeness of the requirements and

5.1.2 Specific requirements are often part of the inspection criteria that drives the inspection process. This presumes an iterative requirements process where inspection results are actionable items that result in appropriate requirements modifications. The Goal-Question-Metric Approach aids in establishing the relative importance of issues and requirements thus helping to determine what needs the rigor of the formal inspection process vs. some other verification process.

5.2 Establish inspection criteria:

Establishing Clear Goals and Decision Points up front provide a focus and milestone schedule for assessing progress and deciding on the future path a project may take. Thus it may address when Formal Inspections should occur since they may be used as key decision points. In addition to requirements documents being the target of a formal inspection, detailed requirements often are part of the inspection criteria – especially those that relate to quality and performance. When a program adopts the Open Systems Approach to development Formal Inspections assess compliance with the appropriate selected Commercial Specifications and Standards. Binary Quality Gates are completion criteria for tasks and are defined at a high degree of granularity so that determining completion is a trivial matter of checking "done" or "not done" rather than making a subjective judgment. Each gate may be comprised of one to many criteria statements. The value of the practice lies in the quality and completeness of the quality gate definitions. Formal Inspections can be applied to improve the quality of those definitions, and then the definitions are used as inspection criteria by the Formal Inspections practice in tracking progress on tasks.

5.3 Manage the inspection process:

Formal Inspections follow a defined process with definite steps, participant roles, entry and exit criteria, and data to be collected (actionable output). They do not fit well with "ad hoc" development. In order for inspections to work project leaders need to provide the appropriate resources, specifically time within the project schedule, and training for participant roles. This is part of the *People Aware Management Accountability* practice. *Configuration Management* is needed to ensure the integrity of the artifacts that are the subject of *Formal Inspection*, and to ensure that inspection results are actually transferred to corrective actions and tracked through completion. This implies that the inspection process would feed into a defect tracking system as part of configuration management. A *Structured Development Methodology* is the natural environment for implementing formal inspections since results of inspections provide input to successive iterations of the product under development. *Formal Inspections* provide objective data to feed *Statistical Process Control* (SPC), which in turn is used to effect software development process improvement. SPC may be applied to the inspection process itself as part of an overall management strategy. When SPC finds that the FI process is "out of control" then the FI process would need to be modified. Applying a formal inspection to a project or product artifact may be part of the criteria for a *Binary Quality Gate*. Thus the two practices are managed in concert because of their complementary relationship.

6. OUTPUTS FROM THE PRACTICE

6.1 Use inspection results for improvement:

Results of FI provide objective data for successive iterations of a product under development focusing on refinement or improvement of the product. If the artifact is an interface design, or an Architecture representation, an iterative cycle, typically associated with a Structured Development Method, enables optimum use of formal inspections results contributing to the refinement of the interface definitions or the architecture. The FI process generates defect data which is used in Defect Tracking against Quality Targets (such as defects per function point) to monitor both process and product improvement. Formal Inspections are valuable tools for assessing compliance to specifications and standards and are thus part of the strategy for Ensuring Interoperability.

6.2 Determine the effectiveness of inspections:

Formal Inspections differ from other types of assessment in the degree to which they employ a rigorous process to the subject artifact, and in the objectivity of data that results. Objective data is necessary for and thus affects the quality of any Statistical Process Control implementation. Formal Inspections can impact the value of *Demonstration-based Reviews* by applying the rigorous process and defined participant roles to the review process relative to what is being demonstrated, contributing to a more focused and effective review. The goal of *Defect Tracking against Quality Targets* is to identify a desired quality target and then seek to achieve it. Defects must be tracked in order to assess progress. Formal Inspections contribute to defect detection but also to defect prevention when applied to artifacts early in the life cycle. Thus they can be a major influence in reducing defects and achieving desired quality goals. *Independent Expert Reviews (such as Software Capability Evaluations (SCEs))*, are conducted to assess the degree to which a project uses "repeatable" defined processes that include process improvement cycles. Implementing *Formal Inspections* throughout a project is one way of achieving that process focus. The training and experience of participants during formal inspections tends to "spill over" into other aspects of the development process contributing to the overall acceptance and cultural migration to a process-oriented approach to software development.

Tab. (1)

Defect Detection Technique	Minimum Value	Most Likely Value	Maximum Value
Design Inspections	25%	57%	84%
Code Inspections	19%	57%	70%

CONCLUSION

In this process through formal inspection we can improve quality of software by detecting and eliminating the defects in the early stages of software product development.

REFERENCES

Journal Papers:

[1] Ackerman et. al., "Software Inspections: An Effective Verification Process," IEEE Software, May 1989.

[2] Fagan, Michael E., "Design and Code Inspections to Reduce Errors in Program Development," IBM Systems Journal, Vol. 15, No. 3, 1976.

[3] Fagan, Michael E., "Advances in Software Inspections," IEEE Transactions on Software Engineering, Vol. SE-12, No. 7, July 1986.

[4] Fagan, M. E., "Advances in Software Inspections" [Fagan 1986], *IEEE Transactions on Software Engineering*, V. SE-12, N. 7, July 1986, pp. 744-751.

[5] Kitchen ham, B., Kitchen ham, A., and Fellows, J., "The Effects of Inspections on Software Quality and Productivity", ICL Technical Journal, [Kitchen ham, et. al. 1986] Month, 1986, pp

[6] Freedman, Daniel P. and Weinberg, Gerald M, "Handbook of Walkthroughs, Inspections, and

Technical Reviews, Evaluating Programs And Projects," 1982.

[7] Jet Propulsion Laboratory, Formal Inspections for Software Development, Rev. E (Training Course), 1990.

[8] Kelly, John C. and Hops, Jonathan, "Software Inspections: An Experience Report" (Draft), January. 26, 1990

Books:

[9] Pressman, R., "Software Engineering: A practitioners Approach (4th Edition)", McGraw-Hill, 1997.

[10] Strauss, S. H. and R. G. Ebenau, "Software Inspection Process", McGraw-Hill, Inc, 1994.

[11] Glib, T. and D. Graham, "Software Inspections", 1993

[12] Kelly, John C., "Original Working Draft of Inspections And Walkthroughs Guidebook," January 1990.

[13] National Aeronautics and Space Administration [NASA1],"Software Formal Inspections Guidebook", NASA-GB-A302. August, 1993.

[14] National Aeronautics and Space Administration [NASA3], "Software Formal Inspections Standard", NASA- STD-2202-93, April 1993.