Comparison of Software Inspections and other Software Quality Assurance Techniques

Shama Shahzadi
Department of Information Technology
University of Lahore
Gujrat Campus, Pakistan
shamapugc@gmail.com

Muhammad Umair
Department of Computer Science
Virtual University of Pakistan
Pakistan
ms150200096@vu.edu.pk

***ABSTRACT*-In this paper, we will analyze and discuss how software inspections overweigh other software quality assurance techniques such as reviews, testing, inspection, etc.This will be done by providing the historical data of companies that used inspections while building their software products and comparing it with the data when software inspections were not used. Our results will show that software inspections reduce the development cost of the software product and also improves the quality of the underlying software. Software inspections, testing, reviews, inspections and other SQA techniques will also be discussed briefly. Besides this, we will also look at why companies still hesitate to use inspections and think that this is an overhead cost. We will try to answer the questions that companies put forward for not using Software inspections.**

KEYWORDS- Inspections, Reviews, Testing

I. INTRODUCTION

The purpose of writing this paper is to compare different software quality assurance techniques. Our analysis will show that software inspections overweigh any other software quality assurance technique in terms of cost, quality, benefit and time. We will start with brief introduction of software quality assurance techniques i.e. Reviews, Testing, Inspections, etc. We will also have a look at the benefits of each. Also, we will look at the team structure for performing each technique. While discussing each software quality technique we will look at the historical data of different companies that how this technique either resulted in benefit or detriment for the company.

Organization of paper follows as under. In section II, we will give a brief overview of related work. Section III, IV, and Vdiscuss brief overview of Reviews, Testing and Inspections respectively. Sub-section of each of above section debates on benefits, importance. Also, we will discuss the team structure for performing underlying software quality assurance technique.Historical data will be presented to see the cost of each technique. In section VI, a comparison will be done between Reviews, Testing and Inspections and we will see how Software Inspections overweigh the benefits as compared to other techniques. In section VII, we will try to answer the questions that companies put forward as the reasons for not using software Inspections. Section VIII discusses the future of Software Inspections.

II. RELATED WORK

Our work for this paper is to provide an analysis of different software quality assurance techniques such as Reviews, Walkthroughs, Testing and Inspections; and show how Inspections overweigh the advantages than any other Software Quality Assurance technique. Before jumping on to analysis of each, we start with discussion on each software quality assurance technique.

III. REVIEWS

Introduced by Fagan at IBM, Review is of one of those quality metrics which that is used to ensure the quality of project deliverable. Review involves examining the software, its documents and its related material to discover areas where software project is lagging behind and quality standards have not been followed. Usually all software documents such as SRS, design document, code, process models, test plans etc. are reviewed during the review process. Reviews ensure that all the documents are conforming to the standards and are complete in all respect.

Whenever problems are found during review process, they are written down and passed on to the person who is responsible for correcting these errors.

REVIEW PROCESS- Review process normally consists of three phases, i.e. Pre-review activities, review meeting and Post-review activities.

Pre-Review activities: Review preparation and review planning is done in this phase. Review planning includes tasks such as setting up a review team, arranging a specific data and time for review and distributing the documents that need review. On the other hand, review preparation activity prepares the participants of the meeting in order to hold effective meeting. Team members usually meet (either formally or informally) in order to get overview of the software to be reviewed. Individuals try to look into the software and its related documents in order to prepare themselves for the review meeting.

Review Meeting: In review meeting, program or document that is being reviewed is put forward before the

other members of the review team. Review meeting must not last more than two hours. Review decisions and actions that need to be taken are recorded.

Post-review activities: The issues raised during the review meeting are addressed. These issues can vary ranging from deficiencies in requirements to coding errors. Management review is followed up with this activity in case if it is to be decided whether more resources are needed to correct the issues raised during review meeting.

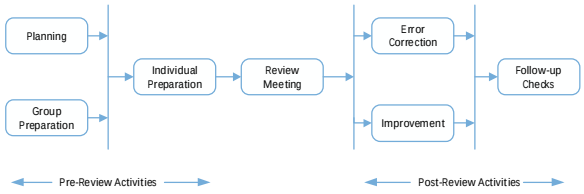

Figure 1 The Software Review Process

TEAM STRUCTURE IN REVIEWS

Review team comprise of 3-4 members who serve as principal reviewers. One of them serves as senior designer who is responsible for all significant decisions during the review meeting. Principal reviewers may also invite other members who are currently working on system development (an example is designer of the system) to contribute to the review. They may not review the whole document but may review that work which is related to them or affects them during development. Project manager is usually not involved in the review meeting, however, he/she will have to take part in the review meeting if some problems require change in the project plan.

REVIEWS: BENEFIT OR DAMAGE?

Whether review meetings should be held or not is still questionable in software engineering domain. Fagan believed that review meeting is crucial as most defects can be identified during review meetings [1]. According to [2], review meetings don't carry a significant impact. On the other hand [3] showed that reviewers were able to find 90% of defects while preparing for the software review meeting whereas only 10% defects were found during software review phase. Other studies such as [4] and [5] also don't speak in favor of review meetings.

IV. TESTING

Software Testing is a formal procedure to detect and then remove the defects that incurred during the development of the software. Testing is usually to demonstrate that developed software meets its requirements. Also, testing tells us that behavior of the system where it does not behave as it would have in normal circumstances.

TESTING PROCESS:

Software Testing Process can be thought of a 5-step process as follows:

- Planning
- Analysis and Design
- Implementation and Execution
- Evaluating exit criteria and Reporting
- Test Closure activities

Now we look at each of them briefly.

Planning: Test Planning refers to those set of activities that are concerned with scheduling and resourcing of the software development. Usually, test plans define what will be tested, methods of testing, and resources available for testing activities. In case of critical systems, test planning may include details of tests that will be run on the software.

Analysis and Design: Test analysis, a process used to derive test information. Test cases will be built to start the testing process. Test Design is a process where inputs are created and their expected outputs are also noted.

Implementation and Execution:

During the implementation step, we create the test data for test cases, whereas during the execution stage, we perform test the software on the test data that we created during the implementation phase. Outcomes of each test case is also recorded after each test.

Evaluating exit criteria and Reporting: Based on the severity level of the project, project manager may specify the exit criteria when defects fall below a specific percentage or number of test cases performed are marked as "enough". Also, project manager may asses if more tests are needed. A summary for stakeholders is also written at this stage.

Test Closure activities: These activities take after tests have been performed. Test closure activities are a mean to check if the planned deliverables are delivered. Test-ware such as test scripts are finalized and handed over to the maintenance team. Also, lessons are learned from the testing so that future projects go better as compared to the underlying project.

TEAM STRUCTURE IN TESTING

Following two levels are often included in software test teams.

Test Manager is responsible for all responsibilities of test planning. He or she is also responsible to check if sufficient resources are available to carry out the testing activities. Test manager also needs to ensure that status report of testing activities is being prepared on regular

basis and if interaction with customers is required on certain cases, then he or she must arrange for it. Test manager is also responsible for updating the project manager with test activities.

Test Engineers, Quality Assurance Testers and Quality Control Testers need to read all the documents so that they understand what needs to be tested and deciding how it will be tested, developing test cases, executing test cases and report about defects along with their priority and severity. When code is changed to fix defects, they are responsible for regression testing.

GUIDING PRINCIPLES FOR TESTING:

For successful software testing, following principles must be followed. [6]

- Start the testing process as early as possible because when you will start early, it will lead to less cost. This will help fix more errors in the early stage of software development. Cost of testing is shown in Figure 2.
- Testing is the process to find defects, therefore, our objective should be to find as many errors as we possibly can, only then true objective of testing can be achieved.
- Complete and to the point requirements are crucial to successful software. Requirements should be clear before testing. Test cases must be written prior to performing testing.
- Valid as well as invalid inputs must be given to the system to perform testing. This will make sure how the system behaves for invalid inputs.
- Errors usually come in clusters. When an error is discovered in a piece of code, probability increases that more errors will be discovered in that code. Therefore, additional testing may be required for such areas.
- Software Testing will not stop. It is a never ending process that has to be stopped somewhere.

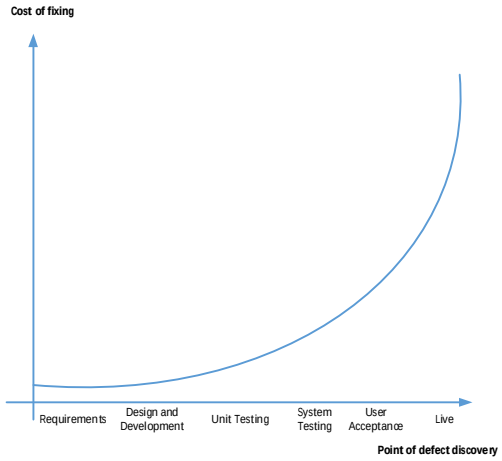

Figure 2Cost of fixing defects

TESTING: BENEFIT OR DAMAGE?

Studies have shown that Software Testing is that step in software development process where most defects are identified. Software Testing adds value to the organization because defects identified during testing are fixed afterwards and therefore quality product is shipped to the customers. Faulty product leads to customer dis-satisfaction and earns less revenue and value for the organization. Also, when customers identify defects while using the product, it takes more time and money to fix defects at that stage. Testing is the only process in software development that makes software useable because it helps identify defects. If defects are not fixed, then customers will stop using the product.

V. INSPECTIONS

Software inspection is a process where team members of software development team meet to find out bugs in the program and where product has not been conforming to the standards. Software inspections can be applied to any complete deliverable or incompletedeliverable of software development process [7]. Software Inspection is somewhat opposite to Software Testing in the way that software does not necessarily needs to be tested during Software Inspections in order to find out bugs. If Testing is not performed then how defects are found? The answer to this question is that a checklist of common bugs to a specific domain are used to find bugs in the program. Also, we can use checklist of different programming errors that occur in a specific language to help identify bugs in the product. Some common type of errors that are looked during inspections are as follows:

- Are values of all variables are initialized before their use?
- Is upper bound of Array is one less than its size?

- Are all conditional statement logically correct? Conditional statement such as `if (a< 15 && a>56)` must not appear in the program.
- Input/output faults should be checked. Examples could be to ensure that all input variables are used, all output variables have a value before they output their value.
- Interface faults, Storage management faults and Exception management faults should also be checked to see if any bugs are contained therein.

TEAM STRUCTURE IN INSPECTIONS

Minimum team size in Software Inspection comprises of three members, however, this number can vary from four to 7. High level documents may require larger teams. Members can be added when their point of view needs to be taken into consideration. Inspector, Moderator, Reader, Recorder and Author play vital role in Software Inspections. We take a look at role of each briefly in the following.

Inspector performs the duty of finding defects in the product. Besides performing duty as an inspector, an inspector may perform duty as moderator, author, reader, and recorder whenever possible.

Usually those personnel are considered good for inspector who have done some similar work in the earlier projects. People who wrote requirements can be a good choice for design inspection. Outside inspectors can also be brought in if they have better knowledge of the system being built.

Moderatoris the one who leads the inspection meeting and ensures that good inspection is carried out. This role is the most critical to formal inspection process therefore, moderators must be trained. Moderator remains active during all inspection activities. Moderator performs duties such as:

- Selecting members for the inspectionteam
- Assigning roles to team members individually and together
- Leading the team throughout the process
- Collecting inspection data on inspection report forms

Readeris responsible for reading the document. He also tries to explain the document with different examples so that everyone in the meeting can understand it clearly and only one interpretation is made by all the participants of everything that he or she reads.

Recorderis responsible for recording the defects and issues that were raised during the inspection meeting. Recorder acts as a reader too as he after recording the defect reads it in the meeting to make sure to everyone that he has written only what was reported and no error has been done by him while recording the defect.

Author creates the underlying work product to be inspected. He or she also answers questions during the meeting so that misunderstanding may not lead to mark something as a defect when actually it isn't. Author also acts an inspector during inspection meeting as he corrects all the defects found during the inspection meeting.

INSPECTION PROCESS

Previous discussion highlighted roles of participants of the inspection meeting. Now we take a look at the inspection process.

Planning: In this phase, participants for the inspection meeting are selected. This selection is done by the moderator. In this step, moderator also takes important inspection material from the author and distributes it to the participants.

Overview Meeting facilitates the moderator in the sense that he uses this meeting to introduce important features of the inspection to the participants.

Preparation allows each member of the team to review its assigned work product. Defects and issues are noted down by him that he will raise during the inspection meeting.

Inspection Meeting is led by the moderator. Recorder records all the defects and open issues that will be raised during the inspection meeting.

Casual Analysis: In this activity, we try to take long term benefits of the inspection process. Casual analysis help us find ways to improve software quality assurance.

Rework: In this activity, issues are settled that were highlighted during the inspection meeting. Issues that couldn't be resolved are held until further meeting or as decided by the moderator.

Follow-up is done to make sure that all the defects that were reported during the inspection meeting have been fixed. If same has been achieved, the inspection meeting is finished.

GUIDING PRINCIPLES FOR INSPECTIONS

For successful software inspections, following principles must be followed. [8]

- Ego should not be brought in inspection meeting. Since producers of the work-product remain an active part of the meeting, it's difficult to put your own work for open criticism. Being a team member, everyone should strive for finding defects before the product is delivered to the customer.
- Criticize the product and not the producer. If someone will start criticizing the producer, then

it will lead to unmanageable behavior during the meeting and the purpose of the meeting will be destroyed.

- Try to find defects during the meeting and never discuss the ways how to fix them.
- Inspection meeting must not last more than two hours as tiredness and boredom becomes obvious after two hours. Long duration will fail the purpose of the inspection meeting.
- Style issues should not be discussed. Style issues could be that someone gives two spaces for indenting the code and someone gives three spaces. These should never be discussed as they are not defects.

INSPECTIONS: BENEFIT OR DAMAGE?

Software inspections have a key benefit that they can be applied earlier in the software project unlike other quality assurance attributes. Since purpose of inspection is to find defects in the work product as early as possible, therefore, earlier defect recovery leads to low cost. Also, software inspections is that quality assurance metric that can be applied on all software development phases i.e., we can apply software inspections on design, requirements, coding, testing, etc. etc. Historical data shows that inspections never lead to negative impact on software products rather high quality software is produced when inspections are used. Figure 3 gives a comparison of the situations when inspections were not used and when inspections were used during software development.

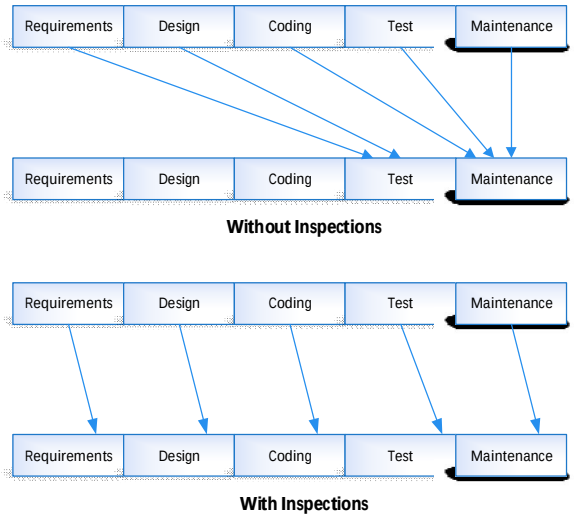

Figure 3Comparison - Inspections vs No Inspections

Above situation is sufficient to believe us that software inspections are very important for developing high quality software. Without inspections, we will end up finding defects in Testing and maintenance phase. On the other hand, when inspections will be used, then we will be able to find defects before leaving that particular life-cycle activity. Defects found early in software work-product incur less cost for correction whereas software defects that are found late incur high development costs. Also, time required to fix such defects is much higher as compared to fixing them when they are found early.

From above discussion, it is clear that benefits of inspections are clear indication that they must be used in software projects. Besides above benefits, inspection help us achieve:

- Schedule deadlines
- Increase customer satisfaction
- Speed up training processes for new products
- Improve development process model
- Team building environment

Since everything has a downside, therefore, benefits don't come free of cost. There is a cost associated with inspections. About 5% to 15% of the total budget is consumed by inspections. However, budget spent on performing inspections is much less than fixing defects at later stage of the software development.

VI. COMPARISON

Now we try to compare the metrics that we discussed in this paper. Table 1 highlights the type of defects that can be found either by inspection or by testing.

Error Type	Inspection	Testing
Module interface errors	✓	
Excessive code complexity	✓	
Unrequired functionality present	✓	
Usability problems		✓
Performance problems	✓	✓
Badly structured code	✓	
Failure to meet requirements	✓	✓
Boundary value errors	✓	✓

Table 1 Comparison of Errors

From above table, it is clear that more errors can be figured out by using Inspections rather than Testing.

If we take a look at Figure 4 (adopted from [9]), then we can clearly see when inspections are not performed then cost to fix defects rises abnormally, whereas when inspections are performed then cost is reduced. Another important thing to be noted here is that when inspections are used, then time to develop the software system is also reduced.

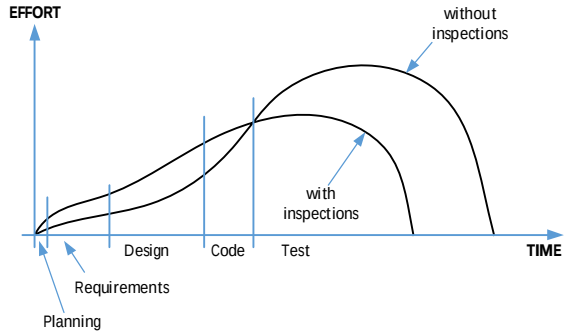

Figure 4With and without inspections

Now, we take a look at the cost of fixing defects in testing phase and compare this with cost of fixing defects with inspections. To do a comparison, we take a look at the historical data of companies. Table 2 shows that data.

Company	Inspections Cost	Test Cost
AT&T	1.40 hrs/defect	8.50 hrs/defect
JPL	$110/defect	$1715/defect
IBM	01 unit of cost	09 times more cost
IBM	$47/defect	$60-$1230/defect
AT&T	01 unit of cost	19 units of cost
ICL	1.2~1.6 hrs/defect	8.39 hrs/defect
Shell	01 unit of cost	29 units of cost
Thorn EMI	01 unit of cost	7~25 units of cost
Applicon, Inc.	01 hour	--
Infosys	01 unit of cost	3 – 6 units of cost

Table 2 Comparison of Inspection cost and testing cost

From the table, we see that inspection cost is much less as compared to cost of fixing defects later in the testing stage.

In this section, we looked at the comparison between software testing and software inspections. We concluded that software inspections overweigh the benefits gained from any other quality assurance technique.

VII. WHY COMPANIES DON'T PERFORM INSPECTIONS?

In the preceding sections, we looked how inspections overweight the advantages when they are used as compared to the situations when we don't use them. Even with these advantages, not too many companies use it. In this section, we take a look at the questions that companies put forward that why they don't use inspections for their software projects. We will also try to answer their questions. Some questions/statements that such companies put forward are:

- Management says that inspections seem to be an added cost
- Inspections are not easy
- Inspection is an unenjoyable task for software engineers
- They are labor-intensive

Now, we try to answer these questions.

- Inspections do take about 5% to 15% budget. However, this cost is reduced cost when seen in terms that when defects will be uncovered during later stages, then they will add more to the cost. Inspections will definitely take money in the beginning, but when a company will be used to inspections, it will realize that inspections save money.
- When something new is adopted, it is difficult. Inspections will be difficult for companies who never used it. But when they will start using inspections in their every software project, they will become expert. Also, new task seems to be boring in the beginning and also requires more work. When companies will become expert in software inspections, then boredom will be diminished.

VIII. CONCLUSION

Software development is becoming a monster with the passage of time because software systems are becoming more complex and unmanageable. To tackle down this monster, we looked how quality assurance attributes can contribute to the quality of software system. We gave analysis and showed how software inspections can be used to build high quality software system. We concluded that those organizations that want to stay in the software industry must take inspections seriously and adopt it in their software projects.

REFERENCES:

[1] Fagan, M.E. (1976). *Design and code inspections to reduce errors in program development, IBM System Journal*, 15(3), 182-211

[2] Johnson, P.M., & Tjahjono, D., (1998). *Does every inspection really need a meeting? Empirical Software Engineering*, (3), 3-35

[3] Eick, S. G., Loader, C.R., Long, M.D., Votta, L.G., & Vander, Wiel, S. (1992). *Estimating software fault content before coding. Proceedings of the 14th Intentional Conference on Software Engineering* (pp. 49-65)

[4] Cheng, B., & Jeffery, R. (1996). *Comparing inspection strategic for software requirement specifications. Proceedings of the 1996 Australian Software Engineering Conference* (pp. 203-211)

[5] Porter, A. A., & Votta, L. G. (1994). *An experiment to assess different defect detection methods for software requirements inspections. Proceedings of 16th International Conference on Software Engineering*, Icse-16 (pp. 103-112)

[6] S.M.K Quadri&Sheikh Umar Farooq, *Software Testing –Goals, Principles, and Limitations, International Journal of Computer Applications* (0975 –8887)Volume 6–No.9, September 2010

[7] *National Aeronautics and Space Administration, Washington, DC 20546,*SOFTWARE FORMAL INSPECTIONS GUIDEBOOK

[8] Karl E. Wiegers, *Improving Quality Through Software Inspections*

[9] Roger S. Pressmanm, *Software Engineering – A Practitioner's Approach*, (pp. 422)