Blekinge Institute of Technology
Software Verification and Validation (PAD001)
Gustav Evertsson, pt99gev@student.bth.se
2002-11-20

Inspection vs. Testing

Introduction

This paper will discuss the pros and cons with inspection and testing and why developers should use them. Both inspection and testing works within the verification and validation area. In some cases they do the same job. They both have the same goal, to raise the quality of the product. Developers could save both time and money and get a better product if they make the right decision. They must know what the strength and weakness in the methods are to make that decision.

It is very important to find the faults as soon as possible because 40 to 50 percent of typical software development effort is devoted to correcting defects. If they find them earlier in the product development there is a lot of time to gain.[1]

Why use inspection?

"Many defects found in testing are directly traceable to requirements and design flaws that could have been detected earlier."[2] Defects that are found later in the project will not only cost more to fix, but also work that are from incorrect design and/or requirements are wasted. Inspection of document is important right from the start of the project. "The objective of these inspections is to find all the defects at each phase and to proceed to the next phase with a completely correct basis."[3]

Inspection can be done in a lot of different ways, different reading methods and so on. The inspection method can have a great impact on how many defects that are found during the inspection. It is important to investigate what method that fits the current project and the company. Some method may take more time to go through but if it is made in the wrong way the result can be worse then if a simpler method is used.

The one that will use the document should make the inspection. This will guarantee that requirements are testable and design is possible to implement. The author of the requirement document must see that everything is included in the document, but may not bee aware that the requirement is the ground to the test specification.

In normal cases are different tests developed to different part of the system. The strengths of inspections are that a similar process can be applied to a wide range of documents.[4] The method can be developed to find more defects earlier and be more effective by using the same method in the whole project or the entire company.

When it comes to finding faults in the code it can be hard to choose between inspection and testing. The right one is depending on what they looking for. It can it be much easier to find faults in code standard and traceability with inspection than it is with testing. Inspection can also be a better help when it comes to debugging and finding what is wrong. This is because inspection will find the fault in the code and testing will only find the failure. According to Sommerville is inspection more effective to find errors. He talks about two reasons way this is true. The first is that inspection can found many defects in a single inspection session. This is because different defects often affect the execution of the program so other defects may be

---

[1] Bill Brykczynski, Reginald Meeson, David Wheeler, Software Inspection: Eliminating Software Defects, p. 2
[2] Bill Brykczynski, Reginald Meeson, David Wheeler, Software Inspection: Eliminating Software Defects, p. 3
[3] Bill Brykczynski, Reginald Meeson, David Wheeler, Software Inspection: Eliminating Software Defects, p. 4
[4] Aybuke Aurum, Håkan Petersson, Claes Wohlin, State-of-the-Art: Software Inspection after 25 Years, p. 5

missed when you run a test. The second reason is that the inspector often have knowledge of the problem and programming language and have seen commonly errors in the code before.[5]

A compiler will strictly follow the logic way of the code but a human will not. He/she will not just follow the logic way of the code as a compiler does. So he can also find errors outside the normal execution. But humans also make mistakes…

Why use testing?

"Testing can consume over 50 percent of software development costs (note that testing costs should not include debugging and rework costs). In one particular case, NASA's Apollo program, 80 percent of the total software development effort was incurred by testing."[6] Some projects can't afford any failures at all during operation like the Apollo project. It can also be that the customers accept some faults that are fixed later under maintenance because the product will be cheaper. So the time spent during testing much depends on what reliability level that are asked for.

One advantage with testing can be that it is closer to the way the end-user will use the system. They will fell that the product has a higher quality because the defects is outside the normal execution. The program will become more reliable by finding the most common failures. Inspection will more look for correctness there a more common executed fault and a more rarely fault is equally easy to find.[7]

I have found in the project that I have been involved in that the test phase often has less priority than other phases. If the project is getting late is it likely that the time for testing will be cut down. This can especially be a problem if the test phase is located at the end of the project and not during the entire project.

Inspection can be effective when same method can be used on a lot of different documents and testing is effective when it comes to rerunning the same test. A good tool for automated testing can take some time to develop but can be executed many times. This can save a lot of time because many systems today are released over and over again. One problem with automated tools is that we need to write additional code and we will not know if the defect is in the systems code or in the test code.

There exist areas where testing is the only option and where you can't find the defects with inspection. Example is tests that test if the system has the right quality attributes with performance and stress testing. But it is also when the developer doesn't have access to the code such as with third party software components and different API's. Different environments outside the program must also be tested, for example running Java applications in different operative systems or applets in different browsers. But there are also things that you can't find with testing such as lack of traceability, design faults etc. It can also be easy to miss faults in code that is not normally executed like exceptions handing.

---

[5] I. Sommerville, Software Engineering, Excerpt: Verification & Validation, p. 426
[6] Gregory T. Daiich, Gordon Priice, Bryce Raglland, Mark Dawood, Software Test Technologies Report, p. 19
[7] David Lorge Parnas, Testing Software

Nothing is perfect

Inspections focuses on finding faults, whereas testing mainly focuses on finding failures (which are the result of one or many faults). They are more compliments to each other than competing methods because they are used to find different faults and in different areas. Developers will probably need to use both inspection and testing to achieve a product with high quality and still be within budget. "Software inspections can identify and eliminate approximately 80 percent of all software defects during development. When inspections are combined with normal testing practices, defects in fielded software can be reduced by a factor of 10."[8] However, researches have shown that the order in which the inspection and the testing are performed will affect the number of defects found[9]. The best way, according to these researches, are to make inspection first and after that the testing. The most popular approach though is the other way around, first testing and then inspection.

Conclusion

What I have found when reading about the two methods is that both is good and must be used to achieve a product that has a high quality and satisfies the end users. They are used for different reasons and in different phases during the project. Each technique has its advantage and way of approaching the search for defects. Their respective strengths help finding different kinds of defects.

Inspections are better for finding errors in design, requirements documents, source code etc. Testing is the only way of finding operational defects, and to make sure that non-functional requirements are working as they are supposed to. Tests can not find errors in requirements documents or in the source code. It depends on previous experience and knowledge about the problem domain among the team members which method that is used. The people performing an inspection may not have the necessary knowledge about the product domain or they may be overloaded with information in the initial stage of the inspection, etc then defects can easily be missed.[10] Testers don't have the same problem because they have test cases to follow.

[8] Bill Brykczynski, Reginald Meeson, David Wheeler, Software Inspection: Eliminating Software Defects, p. 5
[9] Edward F. Weller, Lessons from Three Years of Inspection Data, p 41.
[10] Aybuke Aurum, Håkan Petersson, Claes Wohlin, State-of-the-Art: Software Inspection after 25 Years, p 8.

References

- Bill Brykczynski, Reginald Meeson, David A. Wheeler, Institute for Defense Analyses Software Inspection: Eliminating Software Defects, http://citeseer.nj.nec.com/194919.html

- Gregory T. Daiich, Gordon Priice, Bryce Raglland, Mark Dawood, Software Test Technologies Report, August 1994, http://citeseer.nj.nec.com/daich94software.html

- Aybuke Aurum, Håkan Petersson, Claes Wohlin, State-of-the-Art: Software Inspection after 25 Years, September 2001

- Edward F. Weller, Bull HN Information Systems, Lessons from Three Years of Inspection Data, September 1993

- I. Sommerville, Software Engineering, Excerpt: Verification & Validation, Adison-Wesley, 2000

- David Lorge Parnas, Testing Software, McMaster University, October 8, 1996, http://www.cas.mcmaster.ca/~wmfarmer/SE-2A04-99/testing.slides.pdf