INSPECTIONS

The systematic study of a product artifact (e.g., specification document, design document, source program) for the purpose of discovering flaws in it.

Many studies have shown that:

- Finding errors early in a project costs less to fix than finding errors later in a project.
- The factors of 1 time unit for a specification error, 10 time units for a design error, 100 time units for a coding error, to 1000 time units for an integration testing error.
- These numbers have not been formally validated, but the basic principle that later errors are harder to fix seems to be sound.

All of the following techniques are based on the principle that thinking about an artifact results in better results than simply random testing of the artifact.

READING TECHNOLOGIES

All of the following techniques are an improvement over simply testing a program; however, some are more effective than others. Several related concepts:

Walkthroughs: The developer of the artifact describes its structure at a meeting. Attendees look for flaws in the structure. Weakness – reviewers do not understand deep structure so error finding is weak.

Code reviews: An individual who is not the developer of the artifact reads the text of the artifact looking for errors and defects in structure. Quite effective since reader does not have the same preconceived notion of what the artifact does.

Review: A meeting to discuss an artifact – less formal than an inspection. A traditional checkpoint in software development

INSPECTIONS

Developed by Michael Fagan at IBM in 1972.

Two approaches toward inspections:

- Part of development process – used to identify problems
- Part of quality assurance process – used to find unresolved issues in a finished product

“Fagan inspections” are the former type. Goal is to find defects – A defect is an instance in which a requirement is not satisfied.

Related ideas

Error – a concept that is incorrect.

Fault – the place in the artifact that contains the error.

Failure – the place in the artifact where the error becomes known.

A = 0; ← Fault -- setting of A to 0

B = 10;

C = B/A; ← Failure--Division by A

Error --Division by 0; Why?

FAGAN INSPECTIONS

- Development process consists of series of stages (e.g., system design, design, coding, unit testing, …)
- Develop exit criteria for any artifact passing from one stage to the next
- Validate that every artifact correctly passes the exit criteria before starting on the next phase via an inspection meeting
- Everyone at the meeting must observe that all the exit criteria have been met

INSPECTION PROCESS

Planning – Author gives moderator an artifact to be inspected. Materials, attendees, and schedule for inspection meeting must be set. High level documentation given to attendees in addition to artifact.

Overview – Moderator assigns inspection roles to participants.

Preparation – Artifact given to participants before meeting to allow for study. Participants spend significant time studying artifact.

Inspection – A defect logging meeting for finding defects. (Don’t fix them at meeting; Don’t assign blame; Don’t have managers present; …)

Rework – The author reworks all defects.

Follow-up – Verification by inspection moderator that all fixes are effective.

Observations about inspections

Costly – each participant needs many hours to prepare

Intensive – so limit inspections to no more than 2 hours each

Not for personnel evaluation – limits honesty in finding defects

Cannot inspect too much – perhaps 250 non-commented source lines/hour. More than that causes discovery rate to drop and the need for further inspections to increase.

- Up to 80% of errors found during testing could have been found during an inspection.
- Lucent study (Porter—Votta) Inspection meetings often throw away false positives but find few new errors, so eliminate meeting

Inspection participants

Author – one who developed artifact – does not lead or defend artifact, but controls repair

Reader – one who tries to develop a design of the artifact (Several at meeting.)

Tester – one who tries to test the artifact

Moderator – Conducts inspection; Comes from a neutral organization

Inspections versus Reviews

CONCEPT	REVIEW	INSPECTION
Defects	Opinions of attendees	Violation of specification
Author	Defends position	Observer of proceedings
Error fixing	Discussions at meeting	Later process
Results	List of errors to author	Formal defect list for QA and author
Measurement	None required	Data collected that can be used to develop baseline

PERSPECTIVE-BASED READING

Studied by Basili. Increased error detection if each participant takes on a different role in looking for defects:

Test-Based Reading Questions – Develop tests as if you are in the role of the individual who will test the given artifact:

- Do I have all information necessary to identify the item being tested and identify my test criteria? Can I make up reasonable test cases for each item based upon the criteria?
- Can I be sure that the tests yield the correct answer?
- Do the requirements make sense from what I know about the application or what is specified by the general description?
- Are there other interpretations of this requirement that the implementation might make based upon how the requirement is specified?
- Is there another requirement that would generate a similar test, but yield a different result?

In addition, readers are permitted to address any other defects that they may discover.

User-based reading questions

Develop tests as if you are in the role of the individual who will use the given artifact.

The basic concept is to use scenarios – scripts of sequences of actions the interact with the given artifact being tested.

- Is there any information that prevents you from writing this scenario?
- Are all the necessary functions necessary to write this scenario specified in the requirements, e.g., all the capabilities listed in the general description?
- Are the effects of this scenario specified under all possible circumstances?
- Are the initial conditions for starting up this scenario clear and correct?
- Might some portion of the scenario give different answers depending on how a functional requirement is interpreted?
- Are the interfaces well defined and compatible?
- Do the requirements make sense from what I know about the application or what is specified by the general description?
- Can you get into an improper state, e.g., security or safety problems?

Design-based reading questions

Develop tests as if you are in the role of the individual who will design the given artifact:

- Are all the necessary objects (e.g., data, types, functions) well defined?
- Are all the interfaces defined and consistent?
- Are all the data types compatible?
- Is all the necessary information available to do the design? Are all the conditions involving all objects specified?
- Are there any points that are not clear about what you should do, either because the requirement is not clear or not consistent?
- Is there anything in the requirements that you cannot implement in the design?
- Do the requirements make sense from what I know about the application or what is specified by the general description?

Early Study of PBR

A Study of PBR ["The empirical investigation of perspective based reading" by Basili et al. In *Empirical Software Engineering* vol. 1 (1996) 133-164] shows that:

- 3 readers find more errors than an unstructured reading of a document.
- Two classes of documents read – a "typical" document and a "precise" document. Results more significant in the "formal" document. In the "typical" document, perspectives found fewer unique errors – Why?

Problem with a study like this – Would like to review 2 documents A and B switching technologies. But not possible to do PBR on A and not do PBR on B second. So PBR always second document to read and learning effects cannot be discounted.

Preparing for an inspection

Try to find unique errors – Take on unique role – view artifact from role of designer, tester, user (e.g., Basili's perspective-based reading)

- Write down questions you need to ask author
- Work on artifact at your convenience to find these defects
- Estimate type and severity level of defects found
- Try to describe each defect in a few key words.
- DO NOT TRY AND FIX DEFECT. Fixing on the fly leads to: sloppy fixes and missing many other defects.

Data collect at inspection meeting:

- Preparation time
- Size of artifact inspected
- Number of defects found at each severity level
- Number of unresolved questions to ask author
- Author offline fixes defect, fixes specification, decides problem isn't a defect