Requirements inspections/technical reviews

SEng 321

Outline

- Requirements inspections: integral part of Software V&V activities or Software Quality Assurance (SQA) activities
- Types of technical reviews
- Guidelines for conducting requirements technical reviews (inspections)

A slight detour: Technical Reviews

Technical reviews conducted during:

- specification
- design
- coding

Other names:

- (Fagan) inspections
- Walkthroughs
- Round robin reviews

Benefits of technical reviews

Errors found	Number	Cost unit	Total
Reviews conducted			
During design	22	1.5	33
Before test	36	6.5	234
During test	15	15	315
After release	3	67	201
			783
No reviews conducted			
Before test	22	6.5	143
During test	82	15	1238
After release	12	67	804
			2177

[Roger Pressman, Software Engineering: A practitioner's approach, 1997]

Research (Univ. of Maryland) continued to show since the benefits of technical reviews over automated testing

Characteristics of technical reviews

- Well defined roles
- Typically no more than 6-7 participants
- Duration: no more than 2 hours
- Well-defined and agreed outcome
- Seek to uncover errors close to the time they might have been produced

Roles in technical reviews

Walkthroughs

- Leader
- Recorder
- Reader (implementer)
- Author (designer)
- Other reviewers

Fagan inspections

- Moderator
- Designer
- Coder/implementer
- Tester

Fagan inspection process-- Phases

Planning

Prepare material, educate inspectors, schedule the meeting

Overview and Preparation

Present the overview to participants, read the material to identify defects

Inspection meeting

Find defects! Note but do not solve problems

Process improvement

Learn from current inspection to improve next inspection

Rework

Author reworks all defects

Follow up

Moderator verifies all fixes; product is re-inspect if 5% of document is reworked.

Requirements reviews

Round robin more useful

- Encourages equal participation
- Participants roughly at the same level of knowledge

Guidelines for technical reviews

- Develop a checklist for each work product that is likely to be reviewed.
- Review the product, not the producer.
- Set an agenda and maintain it.
- Limit debate and rebuttal.
- Enunciate problem areas, but don't attempt to solve every problem noted.

Guidelines for technical reviews

- Take written notes.
- Limit the number of participants and insist upon advance preparation.
- Review your earlier reviews and the current review process
- Allocate resources and time schedule for technical reviews
- Conduct meaningful training for all reviewers.

Requirements Document Inspection

Questions to ask as clients…

Why are you looking at the requirements document? We're clients!

What should you be looking for in the requirements document?

What makes a good requirements document?

Concise
Unambiguous
Verifiable
Unique
Identifiable
Traceable
Ranked for importance
Consistent

What makes a good requirement?

Avoid ambiguity (more on this later)
Be careful with names
Write for the reader
Organize carefully
Avoid redundancy (not necessarily repetition)
Avoid unnecessary text
READ WHAT YOU WRITE!

What to look for … as clients.

Pretend you are a TA and mark it :)

Is each requirement stated clearly, concisely, and unambiguously?

Is each requirement testable? How?

Are there conflicting requirements?

What to look for … as clients.

Are the quality requirements stated (and pertinent)?

Are the assumptions and dependencies listed and correct?

Is the scope correct? Or is it insufficient? Or too much?

Does the set of requirements satisfy your needs as a client?

Sample problems

- NFRs/QA that are vague: “the system must be usable”
  - For whom? How usable? What parts?
- “The system should work like the old one”
  - Not possible. The doc should be more specific.
- Spec is full of TLAs and buzzwords
  - Define the Three Letter Acronyms and domain specific terms
- Spec has open questions in improper places.
- ”The elevator must come to the correct floor and then open the doors”
  - This mixes two requirements: the need to come to the floor, and the need to open doors.
- “The system shall offer a Facebook login button”
  - Mixes implementation / design choices. Say “support Facebook login” and let devs figure it out.
- “The main problem for this system is to make people enter more whale data. Whales are important”
  - Not relevant to the requirements. Noise.

Some examples based on https://www.yegor256.com/2015/11/10/ten-mistakes-in-specs.html

What To Do

- Form your group into roles (moderator, lead designer, recorder, tester).
- Go through each of your must-have requirements. Answer the questions on the next page about the requirements.
- Then, look at the design artifacts you have – sketches, data models, and existing code (the designer will have to explain as well). The team-members not moderating or explaining should examine the planned design for potential problems (part 2 of checklist).
- Output: the inspection control sheet, and at least 3 identified problems in the requirements, and 3 problems in the design.