Reviews and Inspections and Quality Management and Agile Dev

Idaho State University | Computer Science

Isaac Griffith

CS 3321
Department of Computer Science
Idaho State University

ROAR

Idaho State University
Computer Science

Topics Covered

- Reviews and inspections
- Quality management and agile development

ROAR

Idaho State University
Computer Science

Reviews and inspections

- A group examines part or all of a process or system and its documentation to find potential problems.
- Software or documents may be ‘signed off’ at a review which signifies that progress to the next development stage has been approved by management.
- There are different types of review with different objectives
  - Inspections for defect removal (product);
  - Reviews for progress assessment (product and process);
  - Quality reviews (product and standards).

Quality reviews

- A group of people carefully examine part or all of a software system and its associated documentation.
- Code, designs, specifications, test plans, standards, etc. can all be reviewed.
- Software or documents may be ‘signed off’ at a review which signifies that progress to the next development stage has been approved by management.

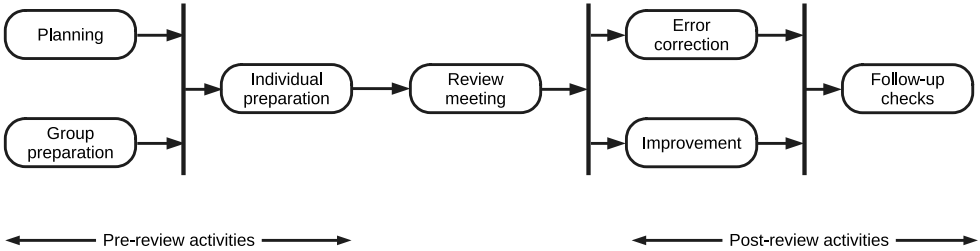
Idaho State University
Computer Science

Phases in the review process

- Pre-review activities
  - Pre-review activities are concerned with review planning and review preparation
- The review meeting
  - During the review meeting, an author of the document or program being reviewed should ‘walk through’ the document with the review team.
- Post-review activities
  - These address the problems and issues that have been raised during the review meeting.

Idaho State University
Computer Science

The software review process



ROAR

Distributed reviews

- The processes suggested for reviews assume that the review team has a face-to-face meeting to discuss the software or documents that they are reviewing.
- However, project teams are now often distributed, sometimes across countries or continents, so it is impractical for team members to meet face to face.
- Remote reviewing can be supported using shared documents where each review team member can annotate the document with their comments.

Program inspections

- These are peer reviews where engineers examine the source of a system with the aim of discovering anomalies and defects.
- Inspections do not require execution of a system so may be used before implementation.
- They may be applied to any representation of the system (requirements, design,configuration data, test data, etc.).
- They have been shown to be an effective technique for discovering program errors.

Inspection checklists

- Checklist of common errors should be used to drive the inspection.
- Error checklists are programming language dependent and reflect the characteristic errors that are likely to arise in the language.
- In general, the ‘weaker’ the type checking, the larger the checklist.
- Examples: Initialization, Constant naming, loop termination, array bounds, etc.

Idaho State University
Computer Science

An inspection checklist

Fault class	Inspection check
Data faults	• Are all program variables initialized before their values are used? • Have all constants been named? • Should the upper bound of arrays be equal to the size of the array or Size - 1? • If character strings are used, is a delimiter explicitly assigned?
Control faults	• For each conditional statement, is the condition correct? • Is each loop certain to terminate? • Are compound statements correctly bracketed? • In case statements, are all possible cases accounted for? • If a break is required after each case in case statements, has it been included?
Input/output faults	• Are all input variables used? • Are all output variables assigned a value before they are output? • Can unexpected inputs cause corruption?

Idaho State University
Computer Science

An inspection checklist

Fault class	Inspection check
Interface faults	• Do all function and method calls have the correct number of parameters? • Do formal and actual parameter types match? • Are the parameters in the right order? • If components access shared memory, do they have the same model of the shared memory structure?
Storage management faults	• If a linked structure is modified, have all links been correctly reassigned? • If a linked structure is modified, have all links been correctly reassigned? • If dynamic storage is sued, has space been allocated correctly? • Is space explicitly deallocated after it is no longer required?
Exception management faults	• Have all possible error conditions been taken into account?

ROAR

Quality management and agile development

Agile quality management

- Quality management in agile development is informal rather than document-based.
- It relies on establishing a quality culture, where all team members feel responsible for software quality and take actions to ensure that quality is maintained.
- The agile community is fundamentally opposed to what it sees as the bureaucratic overheads of standards-based approaches and quality processes as embodied in ISO 9001.

Shared good practice

- **Check before check-in**
  - Programmers are responsible for organizing their own code reviews with other team members before the code is checked in to the build system.
- **Never break the build**
  - Team members should not check in code that causes the system to fail. Developers have to test their code changes against the whole system and be confident that these work as expected.
- **Fix problems when you see them**
  - If a programmer discovers problems or obscurities in code developed by someone else, they can fix these directly rather than referring them back to the original developer.

Reviews and agile methods

- The review process in agile software development is usually informal.
- In Scrum„ there is a review meeting after each iteration of the software has been completed (a sprint review), where quality issues and problems may be discussed.
- In Extreme Programming, pair programming ensures that code is constantly being examined and reviewed by another team member.

Pair programming

- This is an approach where 2 people are responsible for code development and work together to achieve this.
- Code developed by an individual is therefore constantly being examined and reviewed by another team member.
- Pair programming leads to a deep knowledge of a program, as both programmers have to understand the program in detail to continue development.
- This depth of knowledge is difficult to achieve in inspection processes and pair programming can find bugs that would not be discovered in formal inspections.

Pair programming weaknesses

- **Mutual misunderstandings**
  - Both members of a pair may make the same mistake in understanding the system requirements. Discussions may reinforce these errors.
- **Pair reputation**
  - Pairs may be reluctant to look for errors because they do not want to slow down the progress of the project.
- **Working relationships**
  - The pair's ability to discover defects is likely to be compromised by their close working relationship that often leads to reluctance to criticize work partners.

Agile QM and large systems

- When a large system is being developed for an external customer, agile approaches to quality management with minimal documentation may be impractical.
  - If the customer is a large company, it may have its own quality management processes and may expect the software development company to report on progress in a way that is compatible with them.
  - Where there are several geographically distributed teams involved in development, perhaps from different companies, then informal communications may be impractical.
  - For long-lifetime systems, the team involved in development will change. Without documentation, new team members may find it impossible to understand development.

Key points

- Reviews of the software process deliverables involve a team of people who check that quality standards are being followed. Reviews are the most widely used technique for assessing quality.
- In a program inspection or peer review, a small team systematically checks the code. They read the code in detail and look for possible errors and omissions. The problems detected are discussed at a code review meeting.
- Agile quality management relies on establishing a quality culture where the development team works together to improve software quality.

Are there any questions?