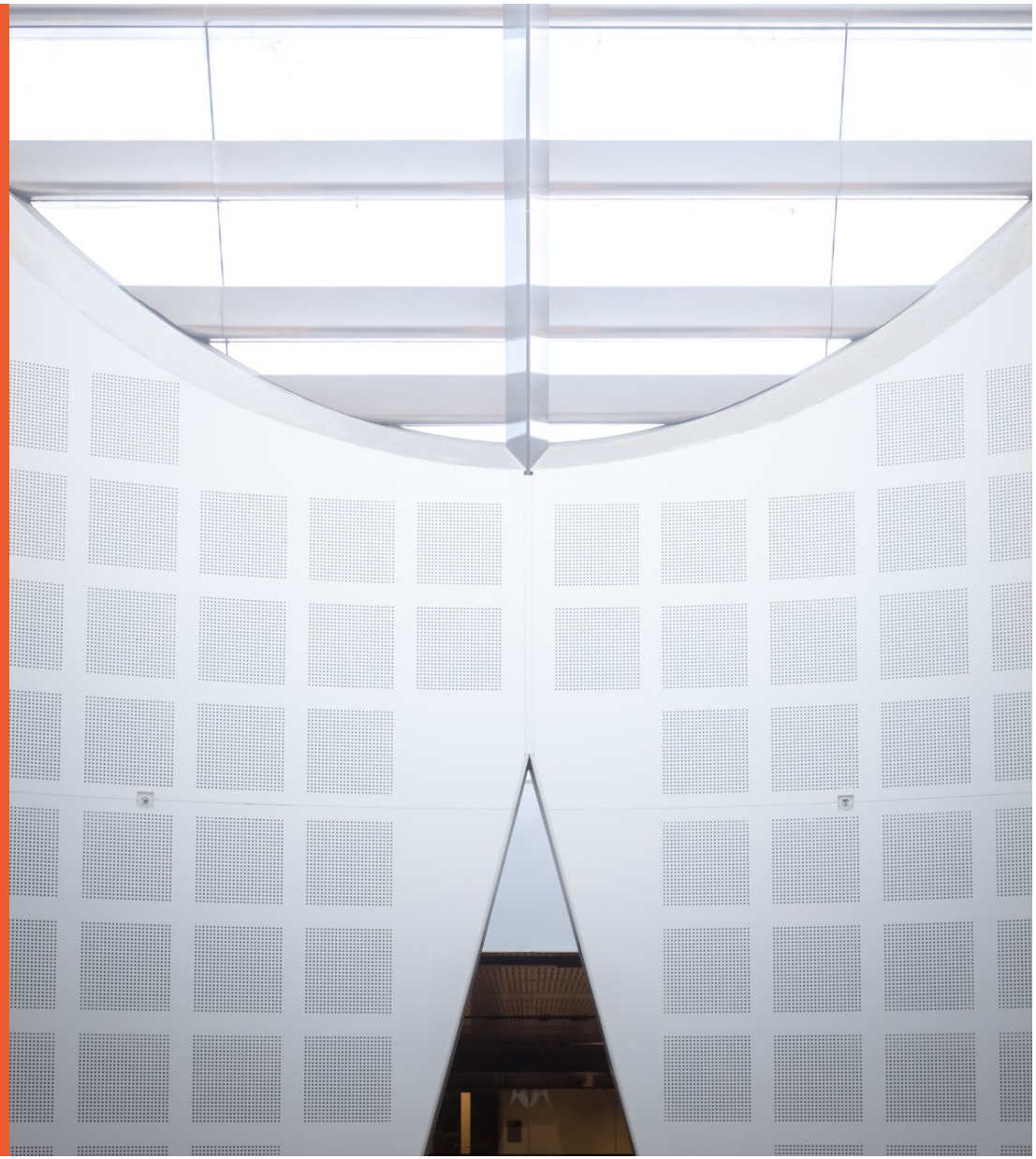
SOFT2201/COMP9201: Software Design and Construction 1

Code Reviews

Dr Xi Wu
School of Computer Science

THE UNIVERSITY OF SYDNEY



Announcement

- We will do an interactive code review session guided by your tutor during this week's tutorial
  - Pair programming will be used during the tutorial. Please have a look at the preview-recommendation page on canvas (under module 8) before you go to your tutorial
- An announcement could be found on Ed regarding the reschedule of tutorials on Thursday (which will be a public holiday)

Outline

- Why review
- What to review
- What is a review
- What is a formal review
- What is an informal review

Why review?

THE UNIVERSITY OF SYDNEY

Why review?

- What is the software, or aspect of the software, to be reviewed for?
  - What are the specifications that need to be met?
- Does the software, or aspect thereof, being reviewed, meet those specifications?

What is the software for and does it work?

- What is the software, or aspect of the software, to be reviewed for?
  - What are the specifications that need to be met?
- Does the software, or aspect thereof, being reviewed, meet those specifications?

- We need a way to check that the software is appropriately designed to meet the expected criteria

What to review?

THE UNIVERSITY OF SYDNEY

What to review?

- Any part of a software project can be reviewed
  - Code
  - Documentation
  - Processes
  - Management
  - Specifications
  - Planning

Status Item	Status up to this week	Planned for next week
Major deliverables		
Major issues		
Major risks		
Estimated effort (hr)		
Status (R, Y, G)		(Not Applicable)

What is a review?

THE UNIVERSITY OF SYDNEY

What is a review?

- A software review is

"A process or meeting during which a software product, set of software products, or a software process is presented to project personnel, managers, users, customers, user representatives, auditors or other interested parties for examination, comment or approval."

IEEE Standard 1028-2008, "IEEE Standard for Software Reviews", clause 3.5

What is a formal review?

THE UNIVERSITY OF SYDNEY

Fagan inspection

- An early effort to formalise the process of reviews
- The basis, or at least similar, to subsequent formal approaches
- Describe program development process in terms of operations
- Define ‘entry’ and ‘exit’ criteria for all operation
- Specify objectives for the inspection process to keep the team focused on one objective at a time

Fagan inspection

- Classify errors by type, rank by frequency, identify which types to spend most time looking for
- Analyse results and use for constant process improvement

Fagan inspection operation

- Specify objectives for the inspection process to keep the team focused on one objective at a time
  - Planning
  - Overview
  - Preparation
  - Inspection
  - Rework
  - Follow-up

Fagan inspection operations

- Planning
  - Preparation of materials
  - Arranging of participants
  - Arranging of meeting place
- Overview
  - Group education of participants on review materials
  - Assignment of roles

Fagan inspection operations

- Preparation
  - Participants review item to be inspected and supporting material to prepare for the meeting, noting any questions or possible defects
  - Participants prepare their roles
- Inspection meeting
  - Actual finding of defect

Fagan inspection operations

- Rework
  - Rework is the step in software inspection in which the defects found during the inspection meeting are resolved by the author, designer or programmer. On the basis of the list of defects the low-level document is corrected until the requirements in the high-level document are met.
- Follow-up
  - In the follow-up phase of software inspections all defects found in the inspection meeting should be corrected. The moderator is responsible for verifying that this is indeed the case. They should verify that all defects are fixed and no new defects are inserted while trying to fix the initial defects.

Formal inspection

- Management review
  - Monitor progress
  - Determine status of plans
  - Evaluate management effectiveness
  - Supports decisions about changes in direction, resource allocation, and scope

Formal inspection

- Management review
  - Maintenance plans
  - Disaster recovery
  - Migration strategies
  - Customer complaints
  - Risk management plans
  - …

Formal inspection

- Management review roles
  - Decision maker
  - Review leader
  - Recorder
  - Management staff
  - Technical staff

Formal inspection

- Management review processes
  - Preparation
  - Plan time and resources
  - Provide funding
  - Provide training
  - Ensure reviews are conducted
  - Act on reviews

Formal inspection

- Technical review
  - Software requirements
  - Software design
  - Software test documentation
  - Specifications
  - … procedures

Formal inspection

- Technical review roles
  - Decision maker
  - Review leader
  - Recorder
  - Technical reviewer

Formal inspection

- Inspections
  - The purpose of an inspection is to detect and identify software product anomalies. An inspection is a systematic peer examination that does one or more of the following:
    - Verifies that the software product satisfies its specifications
    - Verifies that the software product exhibits specified quality attributes
    - Verifies that the software product conforms to applicable regulations, standards, guidelines, plans, specifications, and procedures

External Audits

- Reviews may also be performed by external groups
  - A formal process is generally highly desirable when dealing with external groups and may extend any of the above approaches

What is an informal review?

THE UNIVERSITY OF SYDNEY

Informal reviews

- Reviews may be far less formal
  - Pair-programming
  - “Over the shoulder”
  - Walkthroughs
  - Presentations
  - Self-guided reviews
  - Checklists

Informal reviews

- Formal reviews are far more effective than informal reviews
- Informal reviews are far more convenient than formal reviews

How about a semi-formal code review?

THE UNIVERSITY OF SYDNEY

Purpose

- Have a clear purpose for the review
- Identify all relevant project specifications
- Identify how the specifications can be verified
- Identify who the relevant stakeholders are

Change List/Pull Request reviews

- A complete, ‘working’ project exists
- A set of changes are to be applied
- The changes must be reviewed before they are accepted

Change List/Pull Request reviews

- Owner
  - Is this change wanted?
- Technical reviewer
  - Does this change work?
- Technical reviewer
  - Is this change maintainable?
    - (Can I understand it?)

A brief visit to Design by Contract (DbC)

- A software design approach for program correctness
- Also known as contract programing, programming by contract, design-by-contract programming
- Definition of formal, precise and verifiable interface specification for software components
  - Pre-conditions, postconditions and invariants (contract)

Reviewing DbC change lists

- Definition of formal, precise and verifiable interface specification for software components
  - Pre-conditions, postconditions and invariants (contract)
- Are the pre-conditions met?
- Are the post-conditions met?
- Are the invariants invariant?
- Are these likely to stay so?
- How hard is it to verify?

Change List/Pull Request reviews

- Owner
  - Does the documentation agree with the specifications?
- Technical reviewer
  - Are there tests?
  - Do the tests verify the pre/post conditions are met?
- Technical reviewer
  - Is the code written according to the style guide?
  - Does the code use appropriate design patterns?

Change List review challenges

- Size
  - The more there is to read, the more scope there is to miss problems
  - Many small code reviews are more manageable (somewhat like unit tests vs blackbox system tests)
  - A large code review can take so much time that special planning is needed
  - May need to place limits on acceptable sizes for review

Change List review challenges

- Scope
  - Changes that affect many processes may need more reviewers for the required technical knowledge
  - May sometimes be helped by multiple, smaller changelists
  - May bring specifications from different processes into conflict, requiring management review
    - May also be triggered by small changes with big impacts, such as changing JDK version

Change List review challenges

- Complexity
  - Fast, efficient, code may be hard to read and hard to understand
  - Is the code complexity, thus review complexity and maintenance complexity, worth the improvement?
  - Does the complexity affect maintainability and testability?

Change List review challenges

- Confusion
  - What is the change meant to be for?
  - May be doing too many unrelated things
  - May be making unnecessary changes
  - May even be about a disagreement between colleagues (my approach is better!)

Change List review techniques

- Formalise the review process
  - All changes must be reviewed
  - We already have review roles – Owner, Readability, Technical
  - The planning and preparation are only needed once per project (plus for each major change in direction)
  - The project needs clear specifications, requirements, and sufficient documentation
  - The CL must meet the requirements
  - Changes are recommended and either acted on or disagreed with and the result re-evaluated

Change List review comments

- Be clear
- Be objective
- State the issue
- State what is needed to fix it

- E.g., “Document this method”
- E.g., “These braces aren’t needed, please remove”
- E.g., “Split this file into one per class, for readability”
- E.g., “Use informative variable names”

Change List review examples

- This needs tests
- This needs unit tests
- This needs integration tests
- This needs documentation
- What is this for?
- This doesn’t follow the style guide
- Resubmit without the temp files

Change List review techniques

- Automation is great!
  - When it works
- Code style
- Coverage
- Test suites
- Performance tests
- Spelling (A little more challenging in code)
- Common code issues

Reviewing larger projects

- You may need to review larger projects, rather than changes.
- A formal review process will ensure all parties understand what is expected
- The code inspection will be similar to a changelist review, but with no existing base to compare to
  - More work is needed to verify requirements

Change List review comments

- E.g., “Document this method”
- E.g., “These braces aren’t needed, please remove”
- E.g., “Split this file into one per class, for readability”
- E.g., “Use informative variable names”

- A **report** on a set of changes would discuss why the comments were made, the benefits, the problems, and discuss other approaches
- A **code review** is far more concise and directed

Task for Week 8

- Additional presentation video about code review on project can be found on canvas (Recorded Lecture section)
- Submit weekly exercise on canvas before 23.59pm Saturday
- Attend tutorial on time for an interactive code review session guided by your tutor during tutorial

What are we going to learn on week 10?

- Creational Design Pattern
  - Prototype
- Behavioral Design pattern
  - Memento

Attention Please: due to Monday of week 9 (3 Oct) is a public holiday, we don't have lecture on week 9. BUT we still have tutorials on week 9.

References

- Fagan, M. E. (1976). "Design and code inspections to reduce errors in program development". IBM Systems Journal. 15 (3): 182–211. doi:10.1147/sj.153.0182
- IEEE Std . 1028-1997, "IEEE Standard for Software Reviews“, doi:10.1109/IEEESTD.2008.4601584