Inspecting and Reviewing a Routine, Module or Program

SFWR ENG 2B03

2003

Robert L. Baber

Inspect and Review What?

Object of the inspection and review:

- a routine, module, or program
- against its specification
- and secondarily, the approach taken by the designer and how well the designer executed that approach

I.e. the *Design Report* is the main basis and object of the inspection and review,
but other documentation may also be used.

Primary Goal of Reviewing Software

Either

- confirm correctness, i.e. that the software satisfies its specification and that it is suitable for the intended purpose

or

- identify errors (in the software or in the specification)

Ideally, the first above, but usually the latter

Secondary Goal of Reviewing Software

Identify

- desirable improvements, shortcomings, etc.

e.g. in the interests of

- efficiency (e.g. time, memory, etc.)
- readability, understandability
- systematic structure, simplicity
- modifiability, maintainability
- KISSS

Non-Goal of Reviewing Software

It is *not* the reviewer's job to correct errors or change the program.

That is the designer's job.

The reviewer may and should *suggest* corrections or improvements, if

- appropriate
- little time is required by the reviewer to determine the corrections or improvements.

Methods for Reviewing Software (1)

- mathematical verification
- quantitative, objective measures and characteristics, e.g.
  - nesting depth of loops, if statements
  - length of each procedure
  - variable referenced before value initialized?
  - many other “software metrics”, some helpful, some of controversial value

Methods for Reviewing Software (2)

- qualitative, subjective characteristics, e.g.
  - “readability”, “understandability”
  - program structure (e.g. internal exits from loops, if statements, goto statements)
  - correspondence of code to specification clear? confused?
  - unnecessary repetition?
  - more loops, if statements than necessary?

Methods for Reviewing Software (3)

- qualitative, subjective characteristics, e.g.
  - unnecessarily complicated code? (e.g. if statement instead of simple assignment, confusing nesting)
  - unnecessarily complicated logic?
  - overly complicated, confusing conditions? (e.g. in if statements, loops)
  - modular structure? good subdivisions?
  - constants inappropriately present in program?

Methods for Reviewing Software (4)

- qualitative, subjective characteristics, e.g.
  - side effects?
  - comments in source code appropriate, adequate?
  - loop invariants, intermediate conditions present, appropriate?
- combinations and extensions of all of the above methods

Characteristics of Review Procedure

No matter which methods you use, your inspection and review must be

- systematic: guided by checklists, question lists, etc.
- active: assume nothing, check everything

Remember: You are just as responsible for an error you did not find as the designer is for making it.

Results of Inspection and Review

In your Inspection and Review Report you must at least:

- describe the inspection methods used
- state the specific criteria used
- list the errors identified (if none, justify)
- justify your confidence that no other errors are present
- list other shortcomings, possibilities for improvement

Results of Inspection and Review

In your Inspection and Review Report you should also:

- suggest corrections and improvements as appropriate

But remember the goal:

- find and identify errors, shortcomings
- *not* correct them.

That is the designer's job.

Summary

Inspection and Review of a routine

- identifies errors or confirms their absence (with justification)
- identifies shortcomings, possibilities for improvement
- suggests corrections of errors and improvements as appropriate
- does not correct or change the program code; that is the designer’s job.

References

See

- 2A04 literature, especially slide set 04 “Verification and Analysis”
- 2B03 literature references in Course Outline