

1. What are Code Inspections?

Code inspections are the examination of source code by a suitably qualified group of software engineers with a view to identifying defects in that code.

2. Why perform Code Inspections

- studies have shown that software development cycles which contain Code Inspections prior to the test phase will be shorter than those without Code Inspections
- it is cheaper to locate and fix defects during the coding phase of a development cycle rather than the test phase - this holds true throughout the development cycle; the sooner a defect is detected the less costly it is to fix
- Code Inspections permit project leaders/managers to monitor the performance of the software engineers
- engineers can learn from their peers through Code Inspections

3. How are Code Inspections performed [*PROCEDURE*]

The Fagan technique of Code Inspections is well documented and is summarised below (original reference: *Design and code inspections to reduce errors in program development*, Michael Fagan, IBM Systems Journal No. 3, 1976). As a general rule - when in doubt apply **common sense**.

- project manager appoints a moderator to be responsible for conducting the Code Inspections
- moderator and project manager decide together the percentage of code to be inspected - the more the better. Inspections are time consuming, but time will be saved in later phases of the project
- moderator and project manager draw up a list of code inspectors - inspectors do not have to be members of the project team. The list should contain engineers with various levels of experience
- project manager identifies the code to be inspected
- moderator selects inspectors for a particular section of code - the recommended number is 3 people per section of code
- moderator sets up the inspection meeting, informing both the code inspectors and code author
- moderator circulates hard copies, with line numbers, of the code to be inspected to the inspectors, together with supporting material (design documents, functional and requirements specifications if necessary, coding standards, list of typical coding defects [see ANNEX A], etc). The inspectors mark suspected defects on the code listings which will be retained as part of the

project file

- moderator may schedule a presentation on the code to be inspected, normally by the author. This would normally be required when a particularly complex piece of code is being inspected.
- at the start of the meeting the moderator will
 - ensure the inspectors have inspected the code - reschedule the meeting if they have not
 - appoint a code reader for the meeting
 - appoint a recorder who will record each defect in a defect form
- during the meeting the code is read and the inspectors will identify detected defects. The author can refute the defect, but the moderator must permit only brief discussions on each defect and if no resolution is forthcoming in the meeting then the defect should be recorded and the moderator can attempt to resolve the issue with author and project manager outside the meeting
- the moderator needs to control the meeting and in particular to protect the author
- following the meeting the moderator will provide copies of the defect reports to both author and project manager. Author and project manager will decide the appropriate action for each defect
- moderator needs to follow up to ensure that each defect has at least been dispositioned – if it has been decided not to rectify a defect then that should be noted in the defect form.
- the moderator should produce a report on the meeting identifying # lines inspected, # defects identified, # defects of each type identified etc. The project manager and author should both receive copies of this report
- code to be inspected should compile cleanly and be syntax checked prior to being made available for inspection. Code should not be run prior to inspection
- recommended inspection rate during preparation and actual meeting are X hundreds lines per hour. As a general rule meetings should not inspect more than 300 lines in one sitting
- inspection techniques can also be applied to specifications and design documents

NOTE: In practice, a full-blown procedure involving several persons, as could be inferred from the above, may be expensive in terms of effort and schedule. Therefore, it is recommended to streamline the process as much as possible; see also “8H. ExampleGeneralreviewProcedure”.

ANNEX A: Sample Checklist for Code Inspections

Examine the code for conformance to the Detailed Design (or equivalent), conformance to the coding standards¹, and correctness.

There are various publicly available checklists which could be tailored to an organisation's needs², and then used as an aid to verify that all relevant areas of the code have been examined.

Typical headings to check under are:

- 1. Variable and Constant Declaration Defects (VC)** (e.g. Are there variables that should be constants?)
- 2. Method Definition Defects (FD)** (e.g. Is every method parameter value checked before being used?)
- 3. Class Definition Defects (CD)** (e.g. Can class inheritance hierarchy be simplified?)
- 4. Computation/Numeric Defects (CN)** (e.g. Is overflow or underflow possible during a computation?)
- 5. Comparison/Relational Defects (CR)** (e.g. Are there improper and unnoticed side-effects of a comparison?)
- 6. Control Flow Defects (CF)** (e.g. Can any nested if statements be converted into a switch statement?)
- 7. Input-Output Defects (IO)** (e.g. Have all files been opened before use?)
- 8. Module Interface Defects (MI)** (e.g. Are the number, order, types, and values of parameters in every method call in agreement with the called method's declaration?)
- 9. Comment Defects (CM)** (e.g. Do the comments and code agree?)
- 10. Packaging Defects (LP)** (e.g. For each class: Is it no more than 2000 lines long (Sun Coding Standard)?)
- 11. Modularity Defects (MO)** (e.g. Is there a high level of cohesion within each package?)
- 12. Performance Defects (PE)** (e.g. Can the cost of re-computing a value be reduced by computing it once and storing the results?)

¹ e.g. "Java Coding Style Guide", <http://www.sun.com/software/sundev/whitepapers/java-style.pdf> (looked up on April 4th, 2005)

² e.g. "Java Code Inspection Checklist", <http://iamwww.unibe.ch/~scg/Resources/PSE/PSE2000/WWW/projekthandbuch/codeInspections/JavaInspectionCheckList.pdf> (looked up on April 4th, 2005)