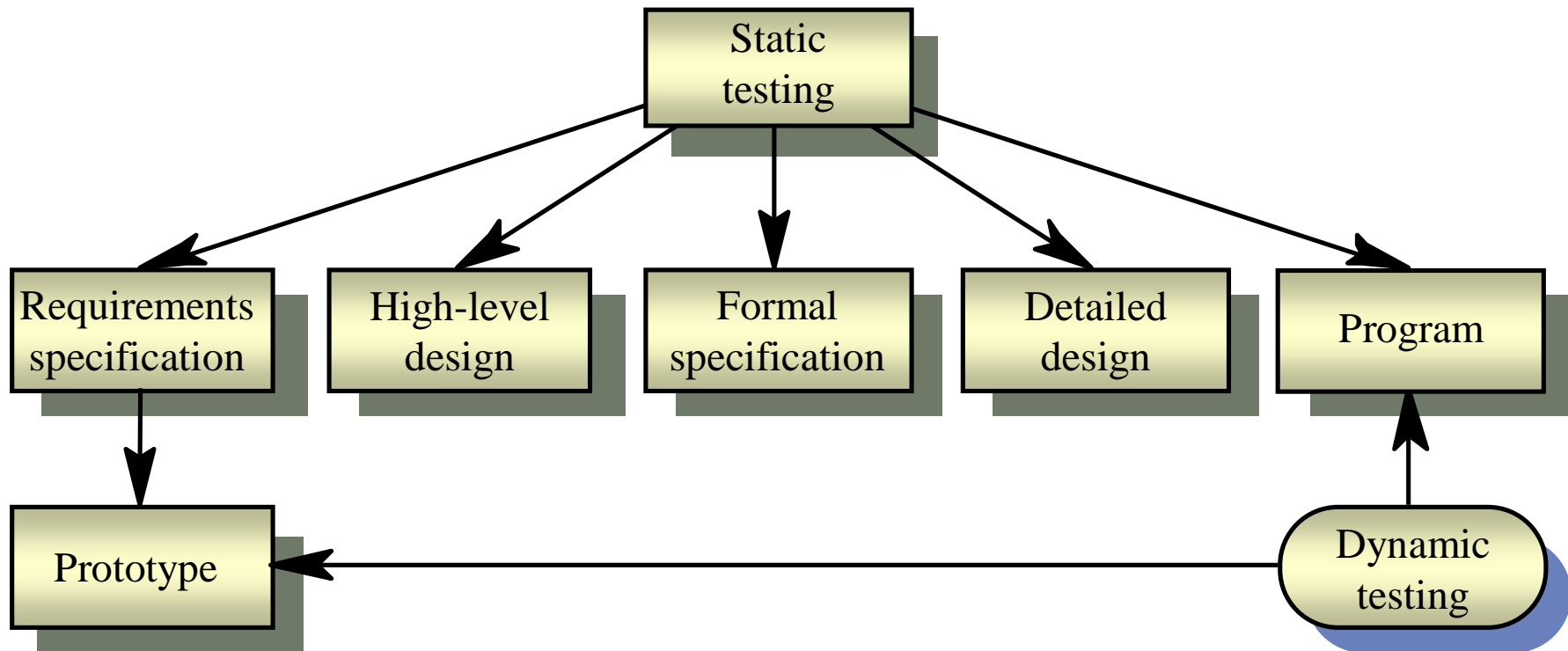


Static and dynamic Testing



Static testing

- ◆ Verifying the conformance of a software system and its specification without executing the code (by a computer)
- ◆ static testing = human testing

Static testing

- ◆ Involves analyses of source text by humans
- ◆ Can be carried out on ANY documents produced as part of the software process
- ◆ Discovers errors early in the software process
- ◆ Usually more cost-effective than testing for defect detection at the unit and module level
- ◆ Allows defect detection to be combined with other quality checks

Static testing effectiveness

- ◆ More than 60% of program errors can be detected by informal program inspections (Meyers: 30 - 70 %)
- ◆ More than 90% of program errors may be detectable using more rigorous mathematical program verification
- ◆ The error detection process is not confused by the existence of previous errors

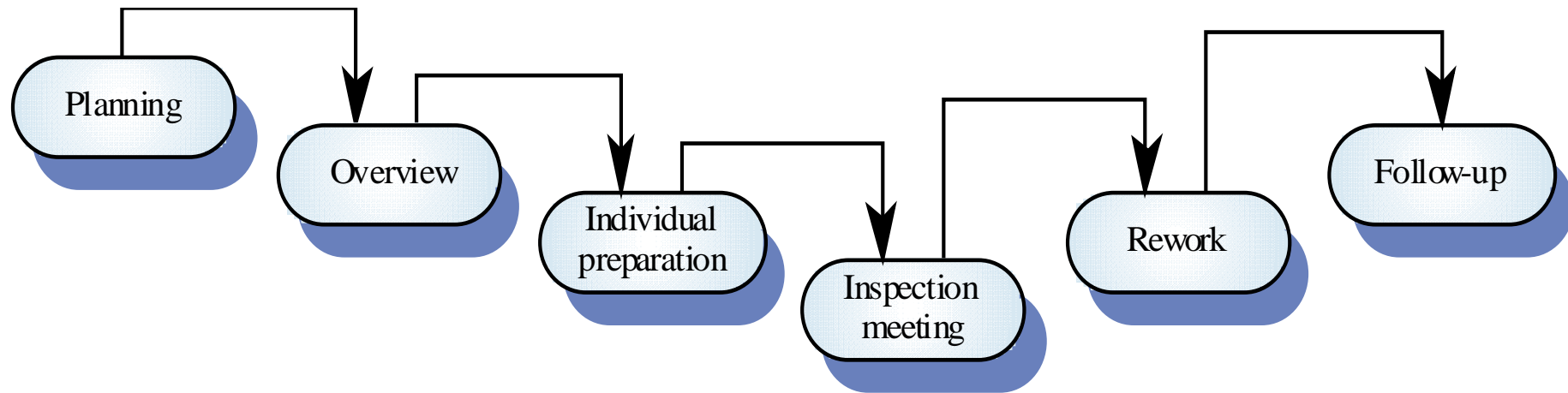
Program inspections

- ◆ Formalised approach to document reviews
- ◆ Intended explicitly for defect DETECTION (not correction)
- ◆ Defects may be logical errors, anomalies in the code that might indicate an erroneous condition (e.g. an uninitialised variable) or non-compliance with standards
- ◆ group code reading → team-based quality

Inspection pre-conditions

- ◆ A precise specification must be available
- ◆ Team members must be familiar with the organisation standards
- ◆ Syntactically correct code must be available
- ◆ An error checklist should be prepared
- ◆ Management must accept that inspection will increase costs early in the software process
- ◆ Management must not use inspections for staff appraisal

The inspection process



Inspection procedure

- ◆ System overview presented to inspection team
- ◆ Code and associated documents are distributed to inspection team in advance
- ◆ Inspection takes place and discovered errors are noted, no repair
- ◆ Modifications are made to repair discovered errors
- ◆ extension of checklists
- ◆ Re-inspection may or may not be required

Inspection teams

- ◆ Made up of at least 4 members
- ◆ Author of the code being inspected
- ◆ Reader who reads the code to the team
- ◆ Inspectors who finds errors, omissions and inconsistencies
- ◆ Moderator who chairs the meeting and notes discovered errors
- ◆ Other roles are Scribe and Chief moderator
- ◆ no superior

Inspection rate

- ◆ 500 statements/hour during overview
- ◆ 125 source statement/hour during individual preparation
- ◆ 90-125 statements/hour can be inspected
Meyers: 150 statements/hour
Balzert: 1 page/hour
- ◆ Inspection is therefore an expensive process
- ◆ Inspecting 500 lines costs about 40 man/hours
effort = £2800

Inspection checklists

- ◆ Checklist of common errors should be used to drive the inspection
- ◆ Error checklist may be programming language dependent
- ◆ Complement static semantics checks by compiler + static analyser
- ◆ The 'weaker' the type checking, the larger the checklist
- ◆ Coding standard / programming guidelines

Inspection checks

Fault class	Inspection check
Data faults	<p>Are all program variables initialised before their values are used?</p> <p>Have all constants been named?</p> <p>Should the lower bound of arrays be 0, 1, or something else?</p> <p>Should the upper bound of arrays be equal to the size of the array or Size -1?</p> <p>If character strings are used, is a delimiter explicitly assigned?</p>
Control faults	<p>For each conditional statement, is the condition correct?</p> <p>Is each loop certain to terminate?</p> <p>Are compound statements correctly bracketed?</p> <p>In case statements, are all possible cases accounted for?</p>
Input/output faults	<p>Are all input variables used?</p> <p>Are all output variables assigned a value before they are output?</p>
Interface faults	<p>Do all function and procedure calls have the correct number of parameters?</p> <p>Do formal and actual parameter types match?</p> <p>Are the parameters in the right order?</p> <p>If components access shared memory, do they have the same model of the shared memory structure?</p>
Storage management faults	<p>If a linked structure is modified, have all links been correctly reassigned?</p> <p>If dynamic storage is used, has space been allocated correctly?</p> <p>Is space explicitly de-allocated after it is no longer required?</p>
Exception management faults	<p>Have all possible error conditions been taken into account?</p>