# Quality Management

**Andy Carpenter**

**School of Computer Science**

(Andy.Carpenter@manchester.ac.uk)

## Software Standards

Define

Attributes of a process

Attributes of a product

International

National

Organisational

Project

## Importance of Standards

Capture wisdom

Framework for defining quality

Avoid previous mistakes

Organization’s view of quality

Assist continuity

Reduce learning curve for new work

Give new staff view of organisation

## Product and Process Standards

- *Product standards*
  - Apply to the software product being developed. They include document standards, such as the structure of requirements documents, documentation standards, such as a standard comment header for an object class definition, and coding standards, which define how a programming language should be used.
- *Process standards*
  - These define the processes that should be followed during software development. Process standards may include definitions of specification, design and validation processes, process support tools and a description of the documents that should be written during these processes.

## Product and Process Standards

| Product standards | Process standards |
|---|---|
| Design review form | Design review conduct |
| Requirements document structure | Submission of new code for system building |
| Method header format | Version release process |
| Java programming style | Project plan approval process |
| Project plan format | Change control process |
| Change request form | Test recording process |

## Product and Process Standards

- Should increase product quality
- Able to be applied and checked cost-effectively
- Within company often derived from international and national standards; e.g.:
  - software engineering terminology,
  - programming language style, notations for drawings,
  - procedures for deriving and writing software requirements,
  - quality assurance procedures
  - software verification and validation techniques

## Problems with standards

- They may not be seen as relevant and up-to-date by software engineers.
- They often involve too much bureaucratic form filling.
- If they are unsupported by software tools, tedious form filling work is often involved to maintain the documentation associated with the standards

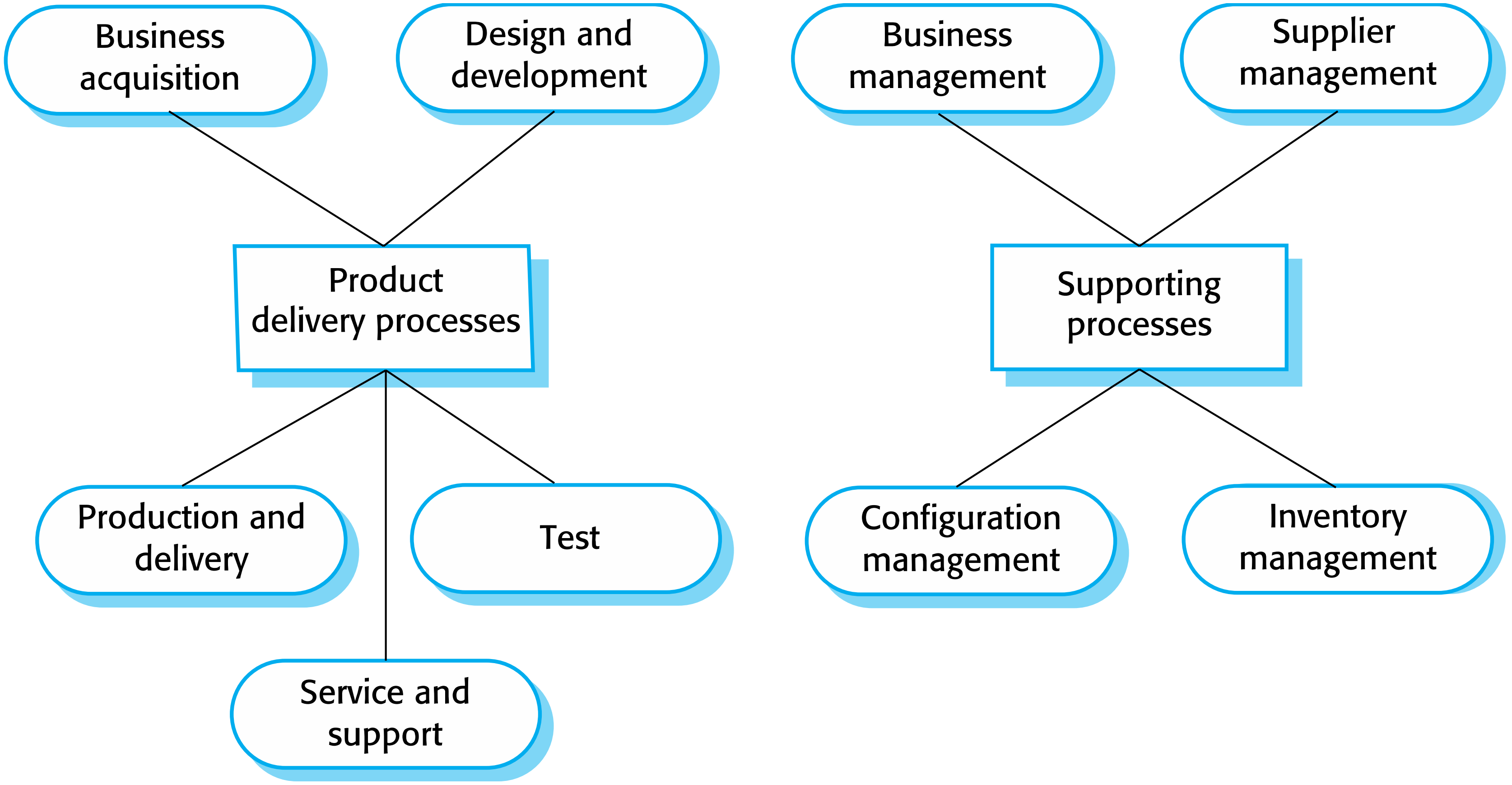
## Standards Development

- Involve practitioners in development. Engineers should understand the rationale underlying a standard.
- Review standards and their usage regularly. Standards can quickly become outdated and this reduces their credibility amongst practitioners.
- Detailed standards should have specialized tool support. Excessive clerical work is the most significant complaint against standards.
  - Web-based forms are not good enough.

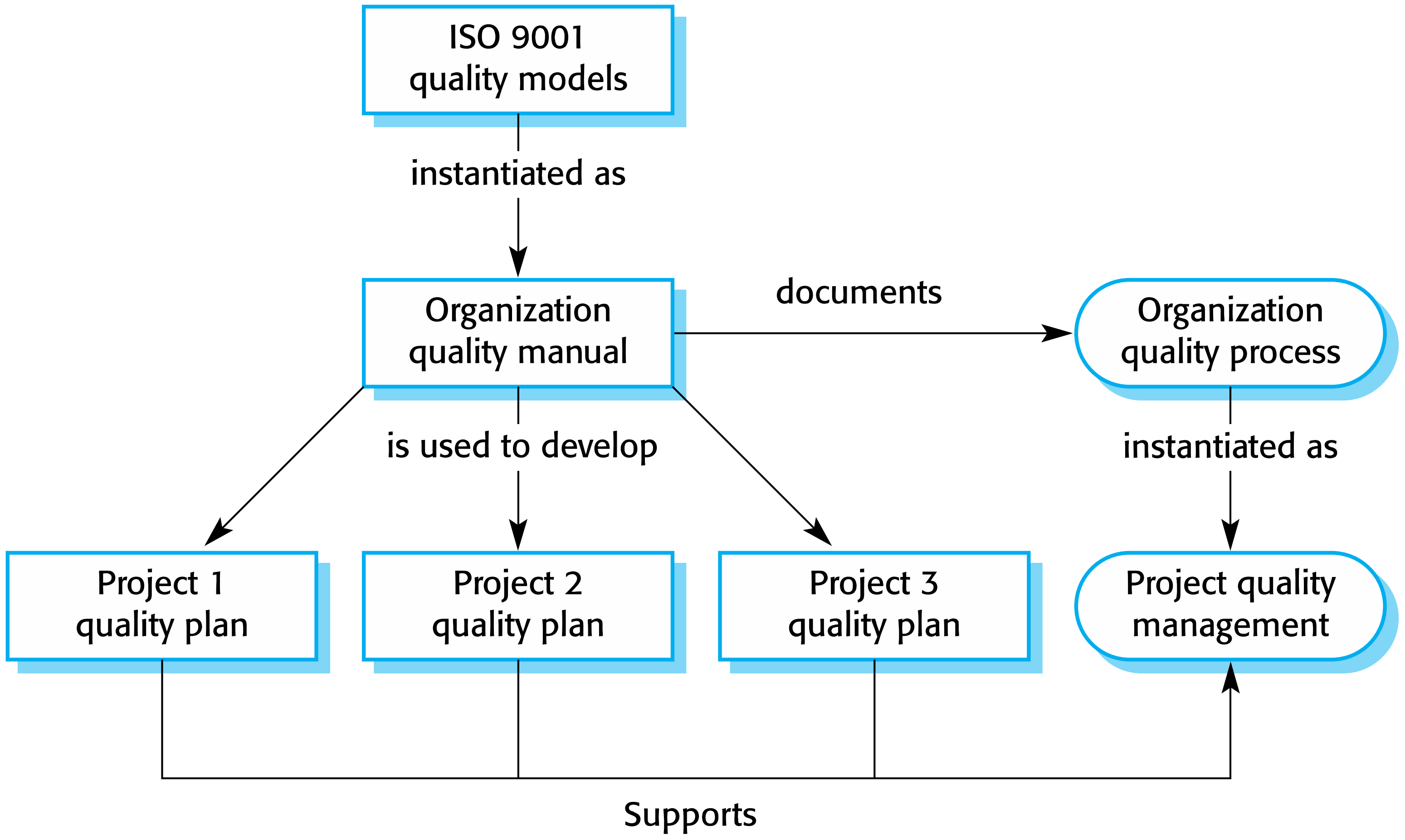
## ISO 9001 standards framework

- An international set of standards that can be used as a basis for developing quality management systems.
- ISO 9001, the most general of these standards, applies to organizations that design, develop and maintain products, including software.
- The ISO 9001 standard is a framework for developing software standards.
  - It sets out general quality principles, describes quality processes in general and lays out the organizational standards and procedures that should be defined. These should be documented in an organizational quality manual.

## ISO 9001 core processes



## ISO 9001 and quality management



## ISO 9001 certification

- Quality standards and procedures should be documented in an organisational quality manual.
- An external body may certify that an organisation’s quality manual conforms to ISO 9000 standards.
- Some customers require suppliers to be ISO 9000 certified although the need for flexibility here is increasingly recognised.

## Software quality and ISO9001

- The ISO 9001 certification is inadequate because it defines quality to be the conformance to standards.
- It takes no account of quality as experienced by users of the software. For example, a company could define test coverage standards specifying that all methods in objects must be called at least once.
- Unfortunately, this standard can be met by incomplete software testing that does not include tests with different method parameters. So long as the defined testing procedures are followed and test records maintained, the company could be ISO 9001 certified.

## Reviews and Inspections

## 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

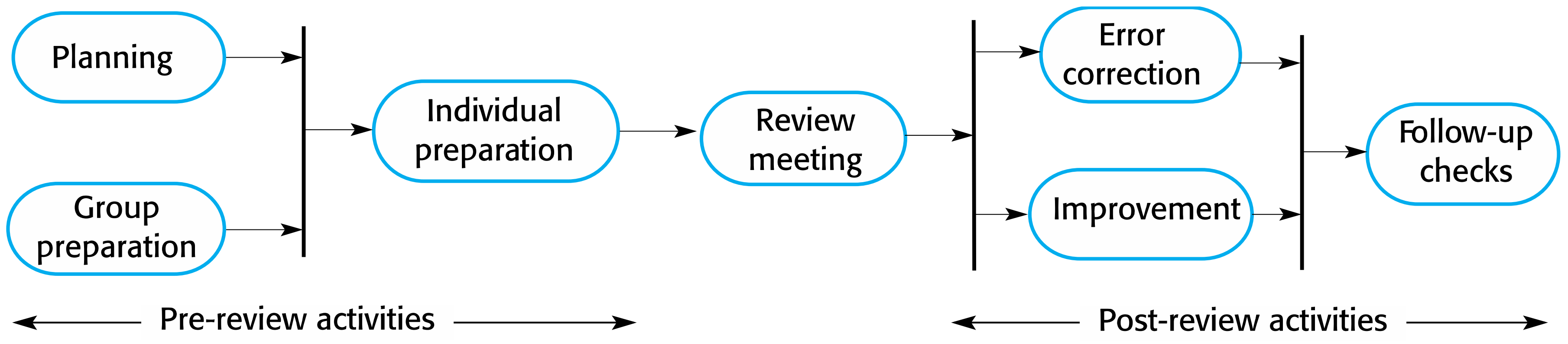
- A group of people carefully examine part or all of a software system and its associated documentation.
- Code, designs, specifications, test plans, standards, etc. can all be reviewed.
- Software or documents may be 'signed off' at a review which signifies that progress to the next development stage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

## 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.

The University of Manchester

## The software review process



## 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: Initialisation, Constant naming, loop termination, array bounds, etc.

MANCHESTER 1824 The University of Manchester

## An inspection checklist (a)

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

# An inspection checklist (b)

| Fault class | Inspection check |
|---|---|
| Interface faults | • Do all function and method calls have the correct number of parameters?<br>• Do formal and actual parameter types match?<br>• Are the parameters in the right order?<br>• 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?<br>• If dynamic storage is used, has space been allocated correctly?<br>• Is space explicitly deallocated after it is no longer required? |
| Exception management faults | • Have all possible error conditions been taken into account? |