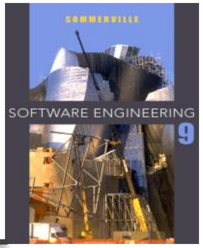
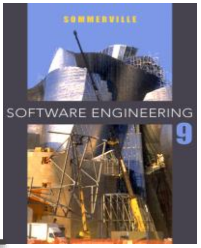


Software Quality Management

PMIT 6111: ST & QA

Software quality management

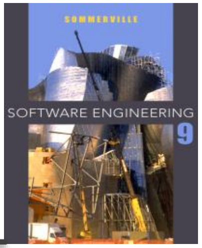


✧ Concerned with ensuring that the required level of quality is achieved in a software product.

✧ Three principal concerns:

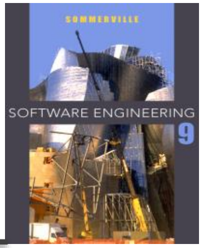
- At the organizational level, quality management is concerned with establishing a framework of organizational processes and standards that will lead to high-quality software.
- At the project level, quality management involves the application of specific quality processes and checking that these planned processes have been followed.
- At the project level, quality management is also concerned with establishing a quality plan for a project. The quality plan should set out the quality goals for the project and define what processes and standards are to be used.

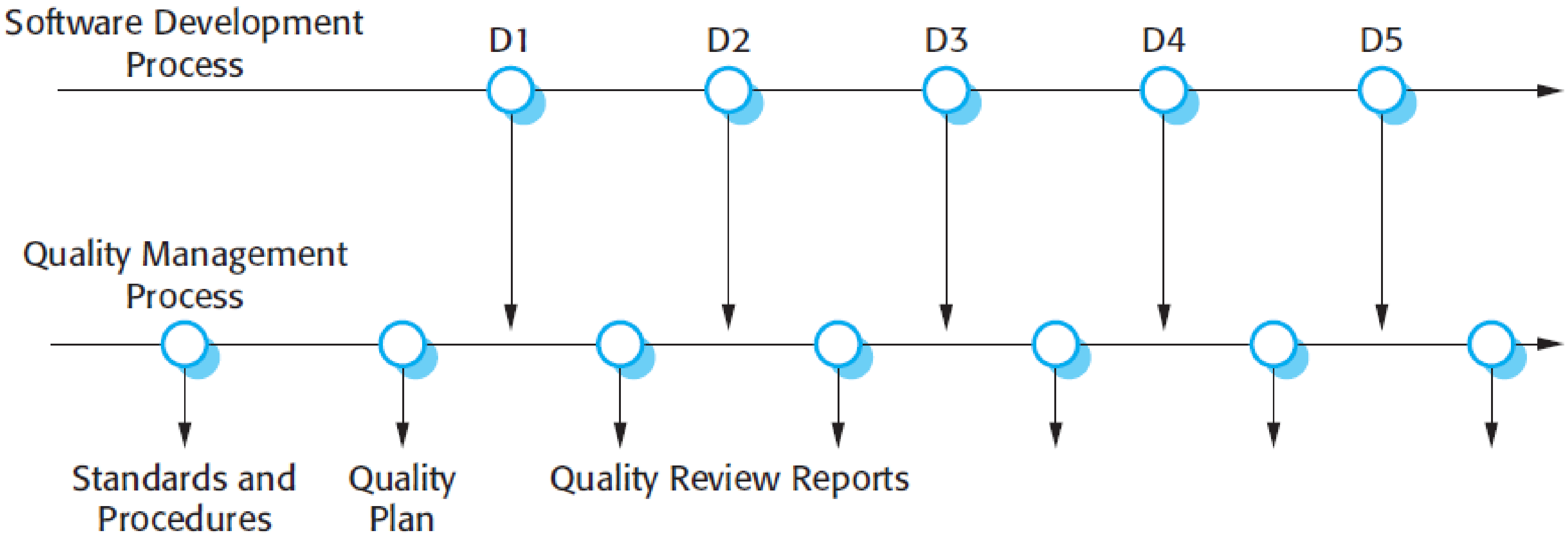
Quality management activities



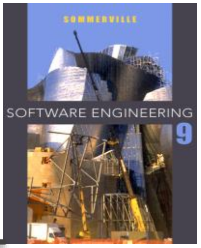
- Quality management provides an independent check on the software development process.
- The quality management process checks the project deliverables to ensure that they are consistent with organizational standards and goals
- The quality team should be independent from the development team so that they can take an objective view of the software. This allows them to report on software quality without being influenced by software development issues.

Quality management and software development



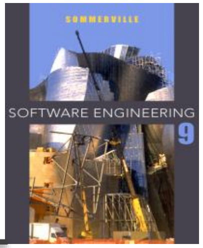


Quality planning



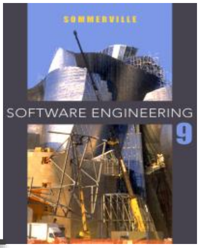
- A quality plan sets out the desired product qualities and how these are assessed and defines the most significant quality attributes.
- The quality plan should define the quality assessment process.
- It should set out which organisational standards should be applied and, where necessary, define new standards to be used.

Quality plans



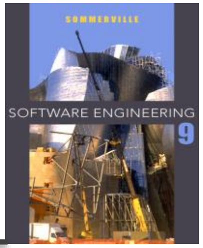
- Quality plan structure
  - Product introduction;
  - Product plans;
  - Process descriptions;
  - Quality goals;
  - Risks and risk management.
- Quality plans should be short, succinct documents
  - If they are too long, no-one will read them.

Scope of quality management



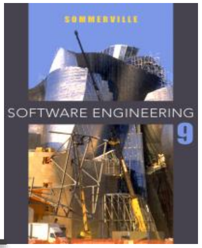
- ✧ Quality management is particularly important for large, complex systems. The quality documentation is a record of progress and supports continuity of development as the development team changes.
- ✧ For smaller systems, quality management needs less documentation and should focus on establishing a quality culture.

Software quality



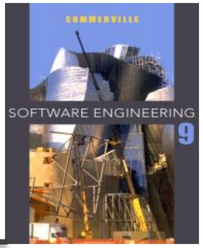
- Quality, simplistically, means that a product should meet its specification.
- This is problematical for software systems
  - There is a tension between customer quality requirements (efficiency, reliability, etc.) and developer quality requirements (maintainability, reusability, etc.);
  - Some quality requirements are difficult to specify in an unambiguous way;
  - Software specifications are usually incomplete and often inconsistent.
- The focus may be ‘fitness for purpose’ rather than specification conformance.

Software fitness for purpose



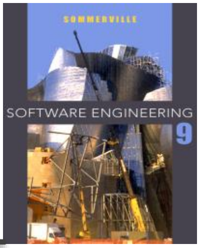
- Have programming and documentation standards been followed in the development process?
- Has the software been properly tested?
- Is the software sufficiently dependable to be put into use?
- Is the performance of the software acceptable for normal use?
- Is the software usable?
- Is the software well-structured and understandable?

Software quality attributes



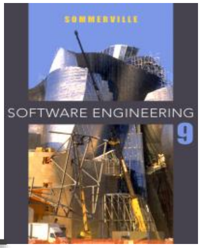
Safety	Understandability	Portability
Security	Testability	Usability
Reliability	Adaptability	Reusability
Resilience	Modularity	Efficiency
Robustness	Complexity	Learnability

Quality conflicts



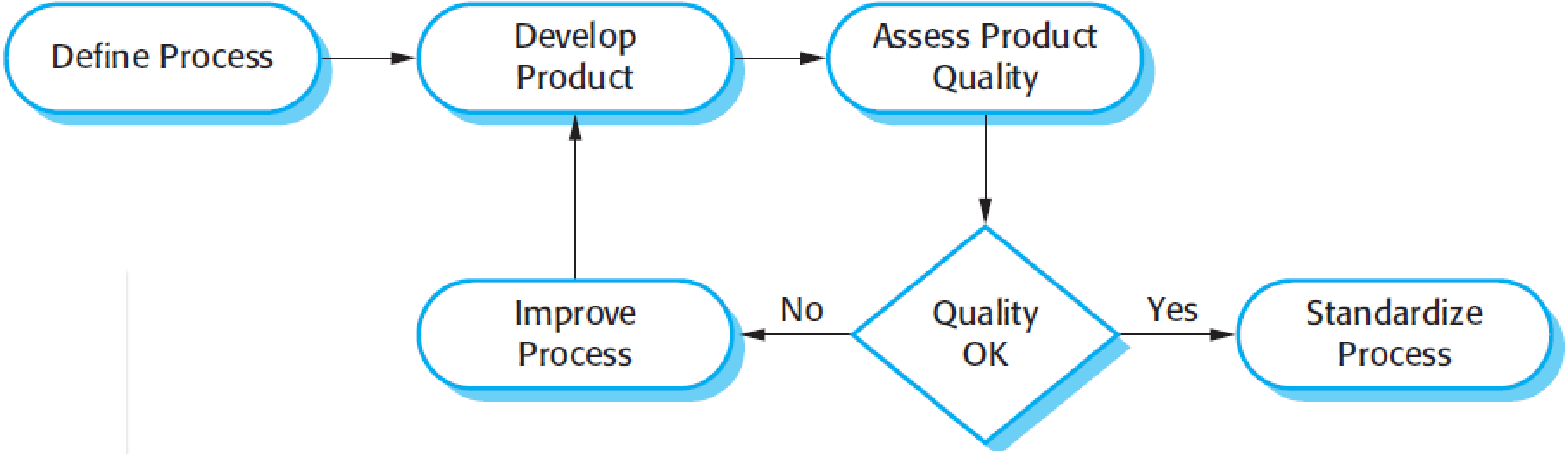
- It is not possible for any system to be optimized for all of these attributes – for example, improving robustness may lead to loss of performance.
- The quality plan should therefore define the most important quality attributes for the software that is being developed.
- The plan should also include a definition of the quality assessment process, an agreed way of assessing whether some quality, such as maintainability or robustness, is present in the product.

Process and product quality

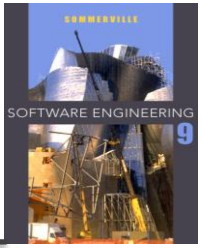


- ✧ The quality of a developed product is influenced by the quality of the production process.
- ✧ This is important in software development as some product quality attributes are hard to assess.
- ✧ However, there is a very complex and poorly understood relationship between software processes and product quality.
  - ▪ The application of individual skills and experience is particularly important in software development;
  - ▪ External factors such as the novelty of an application or the need for an accelerated development schedule may impair product quality.

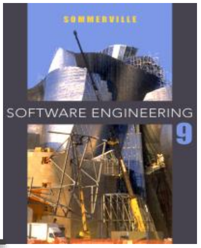
Process-based quality



Software standards



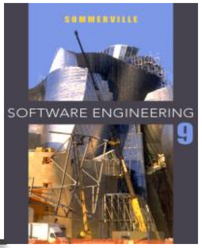
- Standards define the required attributes of a product or process. They play an important role in quality management.
- Standards may be international, national, organizational or project standards.
- Product standards define characteristics that all software components should exhibit e.g. a common programming style.
- Process standards define how the software process should be enacted.



Importance of standards

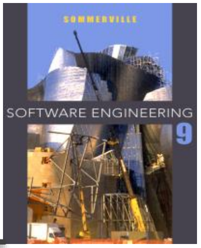
- Encapsulation of best practice- avoids repetition of past mistakes.
- They are a framework for defining what quality means in a particular setting i.e. that organization’s view of quality.
- They provide continuity - new staff can understand the organisation by understanding the standards that are used.

Product and process standards



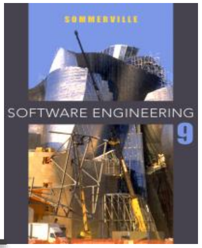
Product standards	Process standards
Design review form	Design review conduct
Requirements document structure	Submission of new code for system building
Method header format	Version release process
Java programming style	Project plan approval process
Project plan format	Change control process
Change request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

Problems with standards



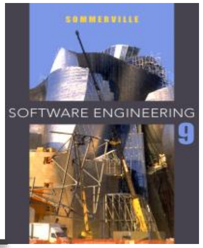
- They may not be seen as relevant and up-to-date by software engineers.
- They often involve too much bureaucratic form filling.
- If they are unsupported by software tools, tedious form filling work is often involved to maintain the documentation associated with the standards.

Standards development



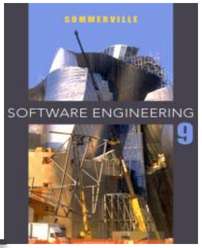
- Involve practitioners in development. Engineers should understand the rationale underlying a standard.
- Review standards and their usage regularly. Standards can quickly become outdated and this reduces their credibility amongst practitioners.
- Detailed standards should have specialized tool support.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

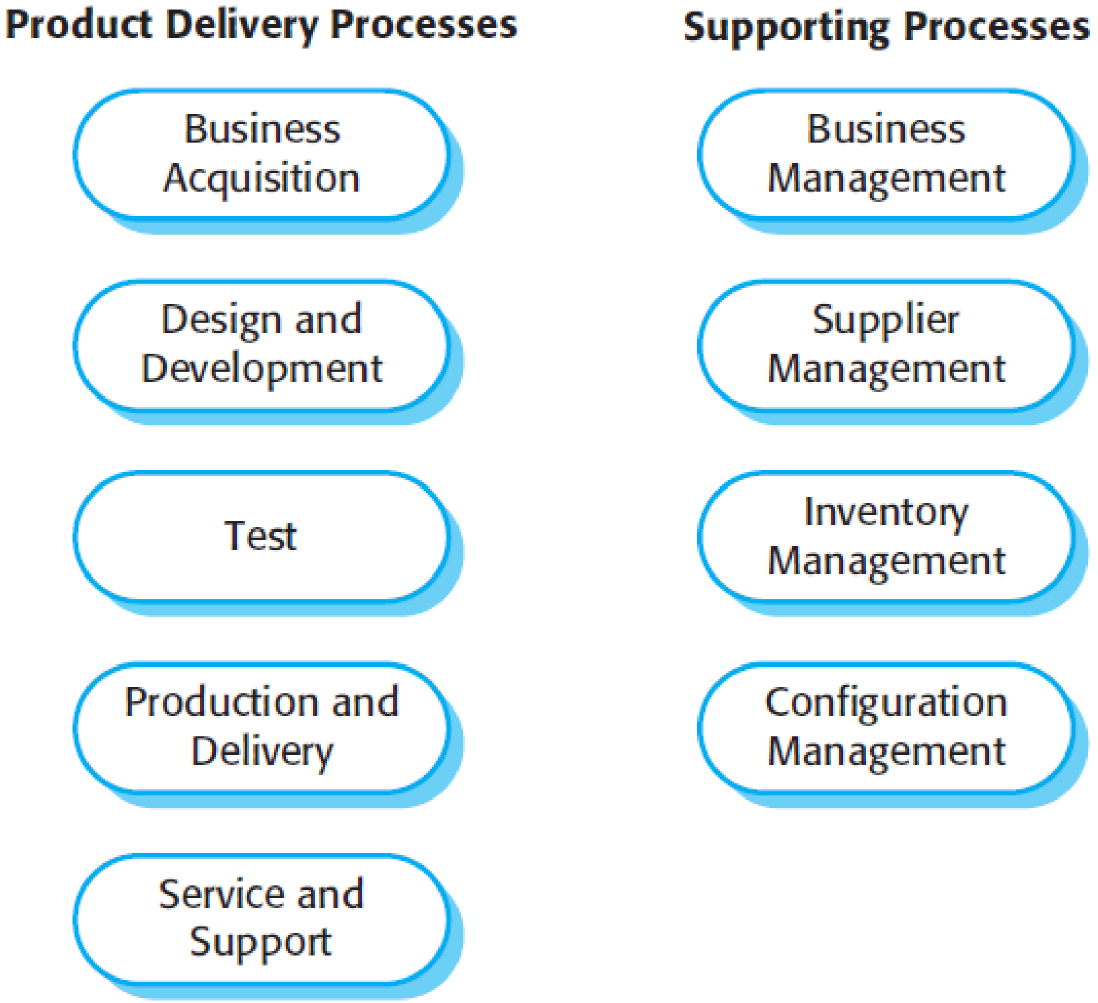
ISO 9001 standards framework



- ✧ 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





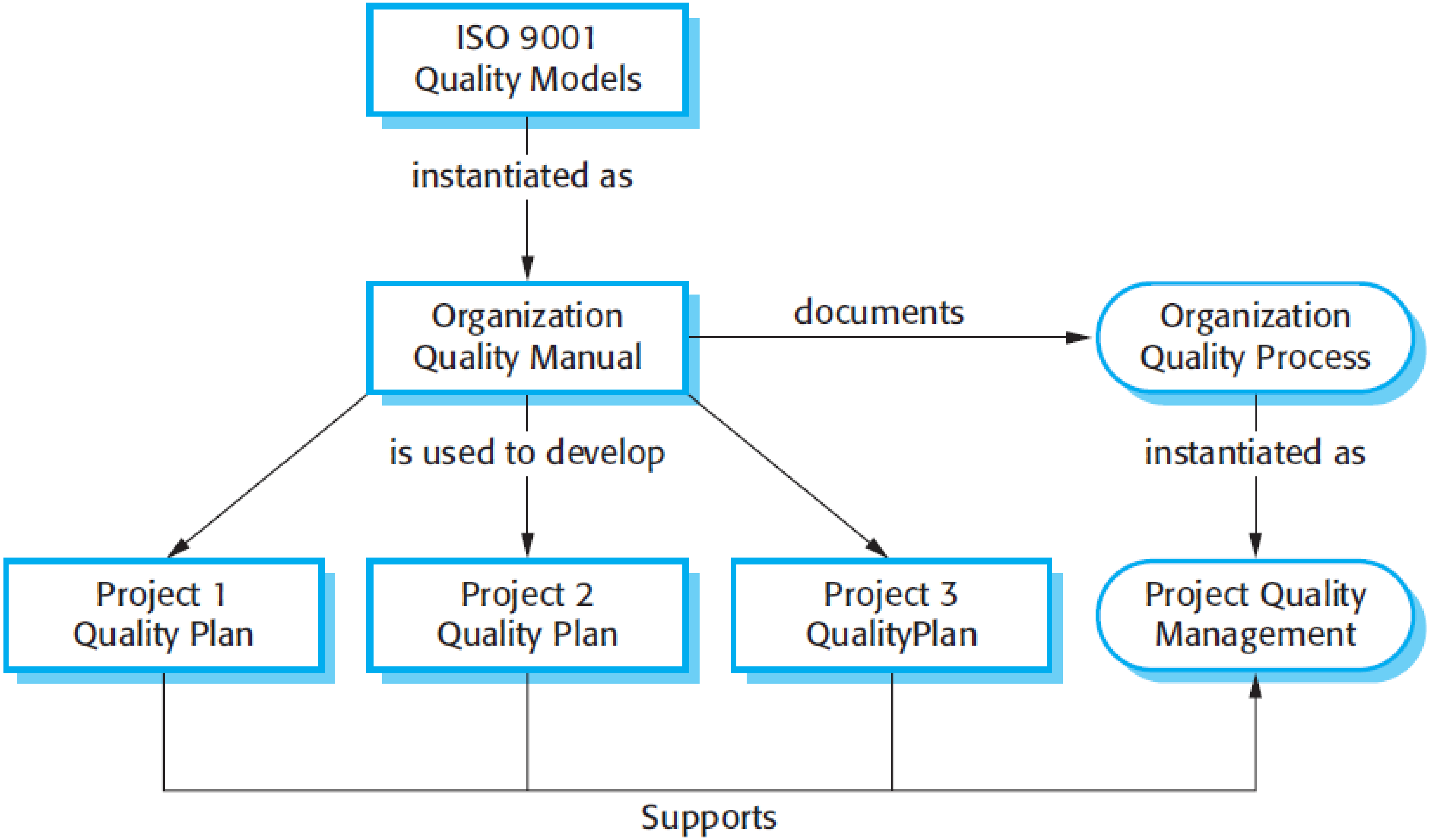
Product Delivery Processes

- Business Acquisition
- Design and Development
- Test
- Production and Delivery
- Service and Support

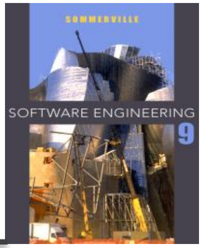
Supporting Processes

- Business Management
- Supplier Management
- Inventory Management
- Configuration Management

ISO 9001 and quality management

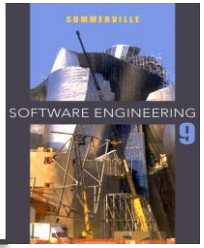


ISO 9001 certification



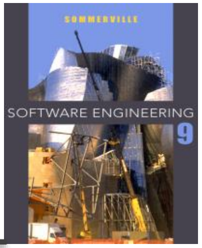
- Quality standards and procedures should be documented in an organisational quality manual.
- An external body may certify that an organisation’s quality manual conforms to ISO 9000 standards.
- Some customers require suppliers to be ISO 9000 certified although the need for flexibility here is increasingly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

Reviews and inspections



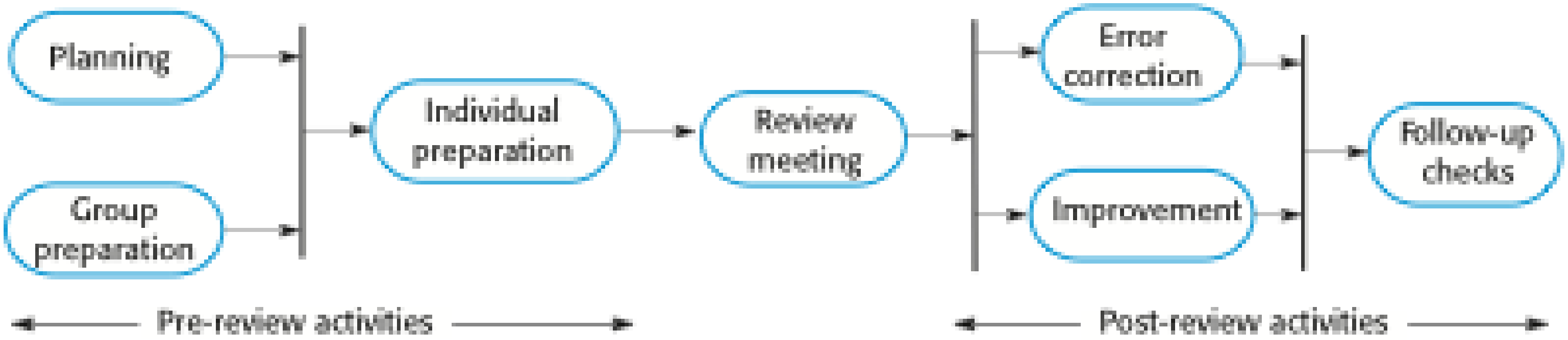
- A group examines part or all of a process or system and its documentation to find potential problems.
- Software or documents may be 'signed off' at a review which signifies that progress to the next development stage has been approved by management.
- There are different types of review with different objectives
  - Inspections for defect removal (product);
  - Reviews for progress assessment (product and process);
  - Quality reviews (product and standards).

Quality reviews

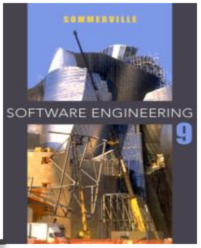


- ✧ A group of people carefully examine part or all of a software system and its associated documentation.
- ✧ Code, designs, specifications, test plans, standards, etc. can all be reviewed.
- ✧ Software or documents may be 'signed off' at a review which signifies that progress to the next development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

The software review process

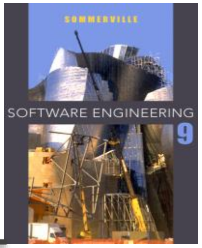


Reviews and agile methods



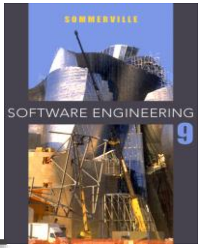
- The review process in agile software development is usually informal.
  - In Scrum, for example, there is a review meeting after each iteration of the software has been completed (a sprint review), where quality issues and problems may be discussed.
- In extreme programming, pair programming ensures that code is constantly being examined and reviewed by another team member.
- XP relies on individuals taking the initiative to improve and refactor code. Agile approaches are not usually standards-driven, so issues of standards compliance are not usually considered.

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



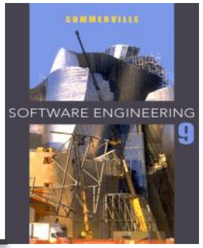
✧ Checklist of common errors should be used to drive the inspection.

✧ 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.

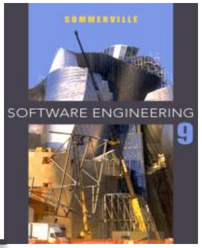
✧ Examples: Initialisation, Constant naming, loop termination, array bounds, etc.

An inspection checklist (a)



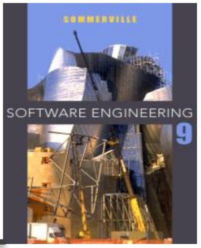
Fault class	Inspection check
Data faults	• Are all program variables initialized before their values are used? • Have all constants been named? • Should the upper bound of arrays be equal to the size of the array or Size -1? • If character strings are used, is a delimiter explicitly assigned? • Is there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

An inspection checklist (b)



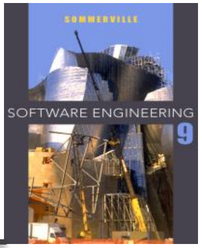
Fault class	Inspection check
Interface faults	• Do all function and method calls have the correct number of parameters? • Do formal and actual parameter types match? • Are the parameters in the right order? • If components access shared memory, do they have the same model of the shared memory structure?
Storage management faults	• If a linked structure is modified, have all links been correctly reassigned? • If dynamic storage is used, has space been allocated correctly? • Is space explicitly deallocated after it is no longer required?
Exception management faults	• Have all possible error conditions been taken into account?

Agile methods and inspections



- Agile processes rarely use formal inspection or peer review processes.
- Rather, they rely on team members cooperating to check each other’s code, and informal guidelines, such as ‘check before check-in’, which suggest that programmers should check their own code.
- Extreme programming practitioners argue that pair programming is an effective substitute for inspection as this is, in effect, a continual inspection process.
- Two people look at every line of code and check it before it is accepted.

Reference



- ✧ Chapter 24: Quality Management, “Software Engineering” by Ian Sommerville, 9th edition.