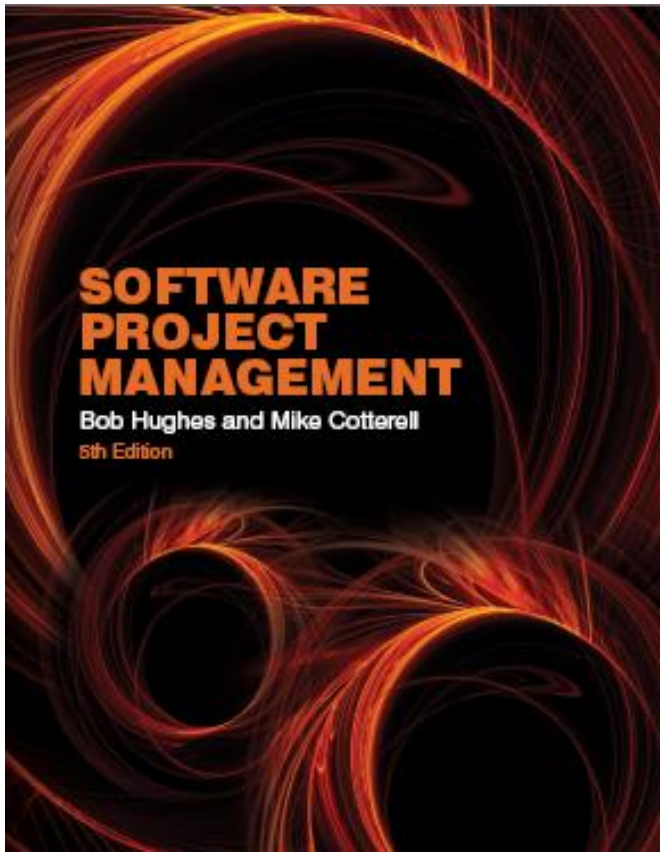
Software Project Management
Fifth Edition



Chapter 13.2

Software process quality

BS EN ISO 9001:2000 and quality management systems

- ISO 9001 is one of a family of standards that specify the characteristics of a good quality management system (QMS)
- Can be applied to the creation of any type of product or service, not just IT and software

BS EN ISO 9001:2000 and quality management systems

- Does NOT set universal product/service standards
- DOES specify the way in which standards are established and monitored

ISO 9001:2000 principles

1. Understanding the requirements of the customer
2. Leadership to provide unity of purpose and direction to achieve quality
3. Involvement of staff at all levels
4. Focus on the individual processes which create intermediate and deliverable products and services

ISO 9001:2000 principles

5. Focus on interrelation of processes that deliver products and services
6. Continuous process improvement
7. Decision-making based on factual evidence
8. Mutually beneficial relationships with suppliers

ISO 9001:2000 principles

- These principles are applied through cycles which involve the following activities

1. Determining the needs and expectations of the customer;

2. Establishing a *quality policy*, that is, a framework which allows the organization's objectives in relation to quality to be defined;

ISO 9001:2000 principles

3. Design of the *processes* which will create the products (or deliver the services) which will have the qualities implied in the organization's quality objectives;

ISO 9001:2000 principles

4. Allocation of the responsibilities for meeting these requirements for each element of each process;

5. Ensuring resources are available to execute these processes properly;

ISO 9001:2000 principles

6. Design of methods for measuring the effectiveness and efficiency of each process in contributing to the organization's quality objectives;

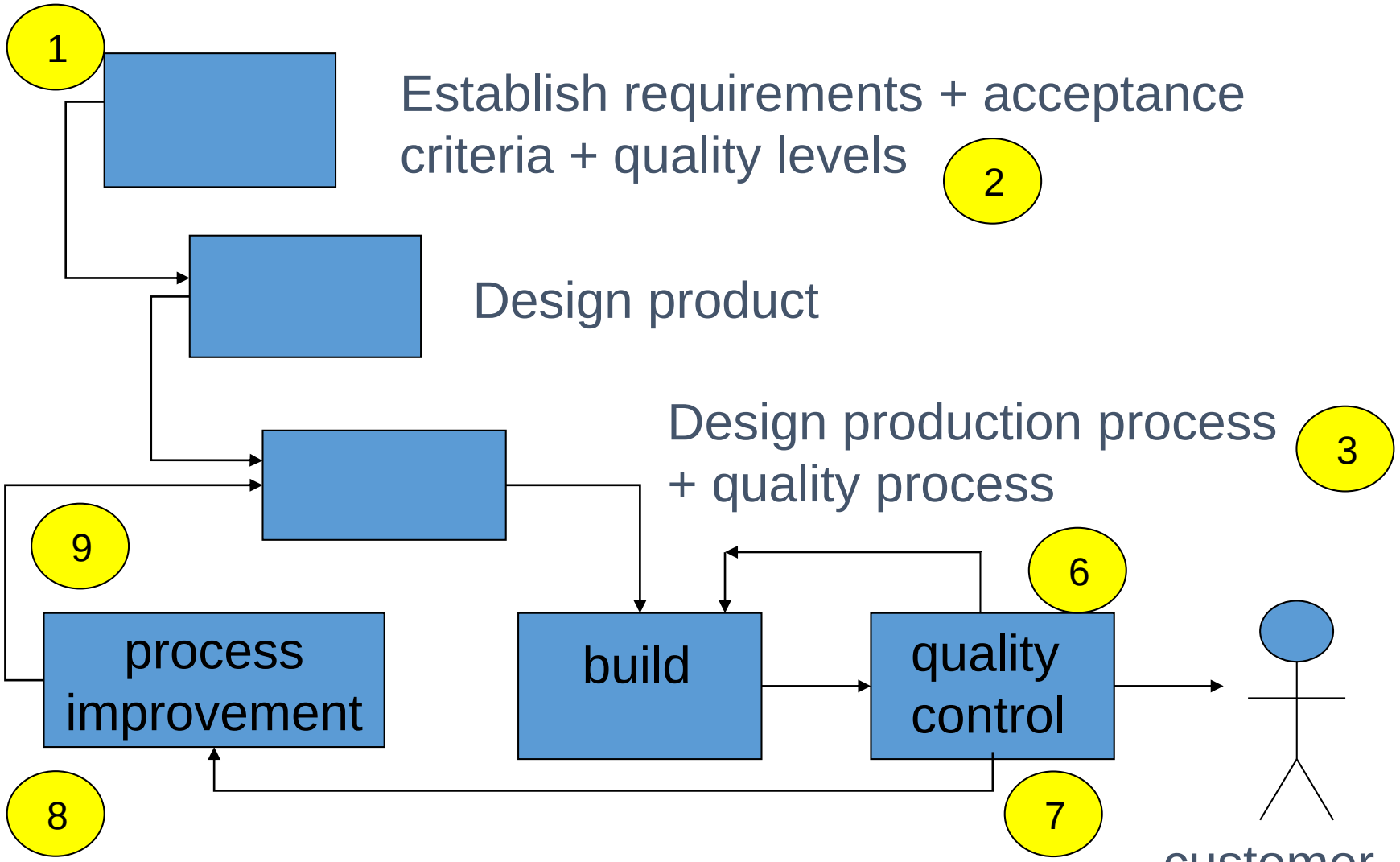
7. Gathering of measurements;

ISO 9001:2000 principles

8. Identification of any discrepancies between the actual measurements and the target values;

9. Analysis and elimination of the causes of discrepancies

ISO 9001:2000 cycle



The need to improve

- Can everything be improved at one?

  no, must tackle the most important things first
- 'Poor companies are poor at changing'

  some later improvements build on earlier ones
- But there are problems
  - *improvement takes up time and money*
  - *'improvement' may simply be more bureaucracy!*

Capability maturity model

- Created by Software Engineering Institute, Carnegie Mellon University
- CMM developed by SEI for US government to help procurement
- Watts S. Humphrey ‘Managing the software process’ Addison Wesley
- Assessment is by questionnaire and interview

Capability maturity model 2

- Different versions have been developed for different environments e.g. software engineering
- New version CMMI tries to set up a generic model which can be populated differently for different environments

CMMI

- Organization is placed at one of the five levels of process maturity
  - Level 1 : Initial – The procedures tend to be haphazard
  - Level 2 : Managed – Basic project management procedures in place, however, the way individual tasks are carried out will depend largely on the person doing it

CMMI

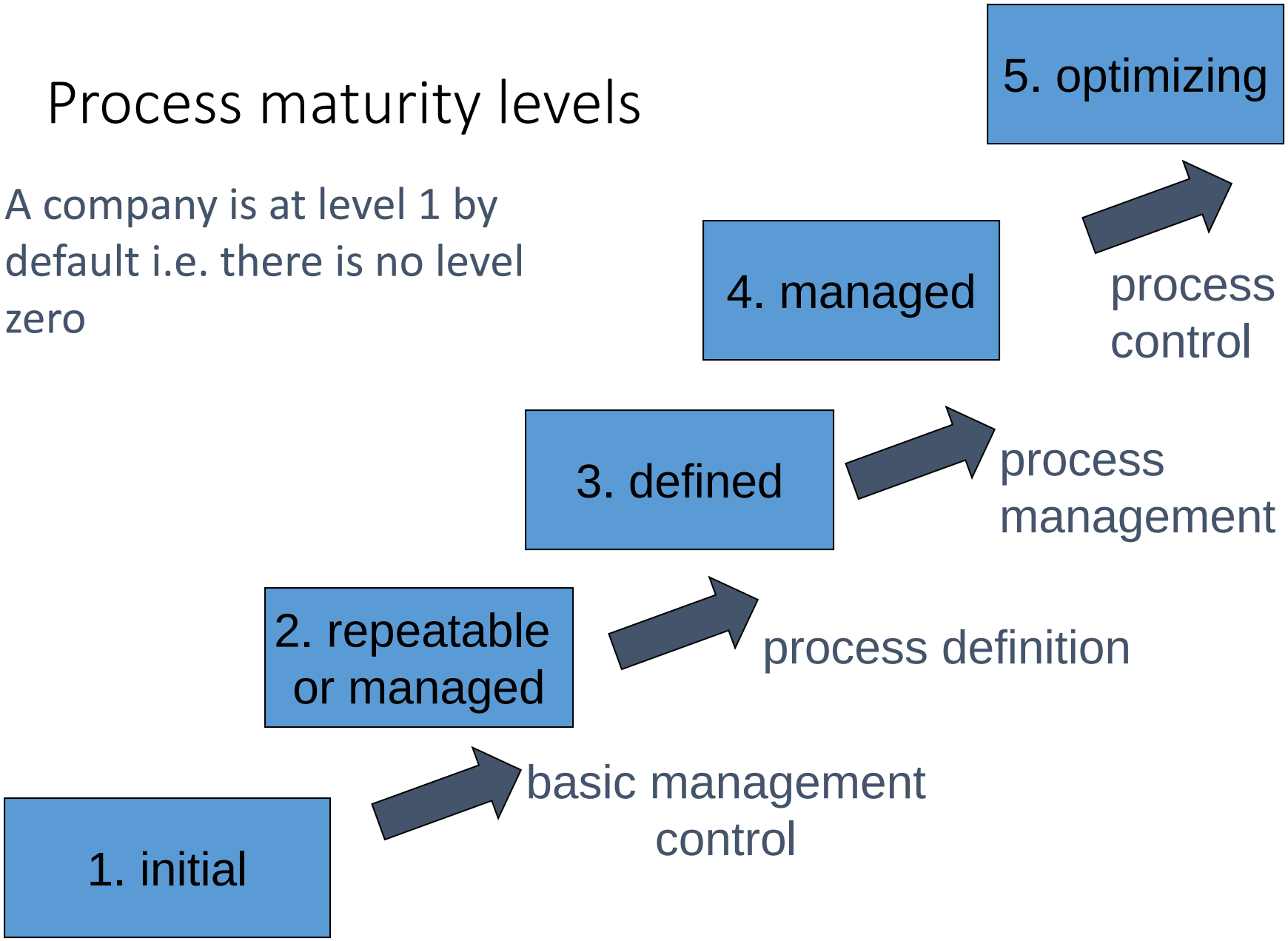
- Organization is placed at one of the five levels of process maturity
  - Level 3 : Defined – The way each task in SDLC should be done is defined
  - Level 4 : Quantitatively managed – Products and processes involved in software development are subject to measurement and control

CMMI

- Organization is placed at one of the five levels of process maturity
  - Level 5 : Optimizing – Improvement in procedures can be designed and implemented using the data gathered from the measurement process

Process maturity levels

A company is at level 1 by default i.e. there is no level zero



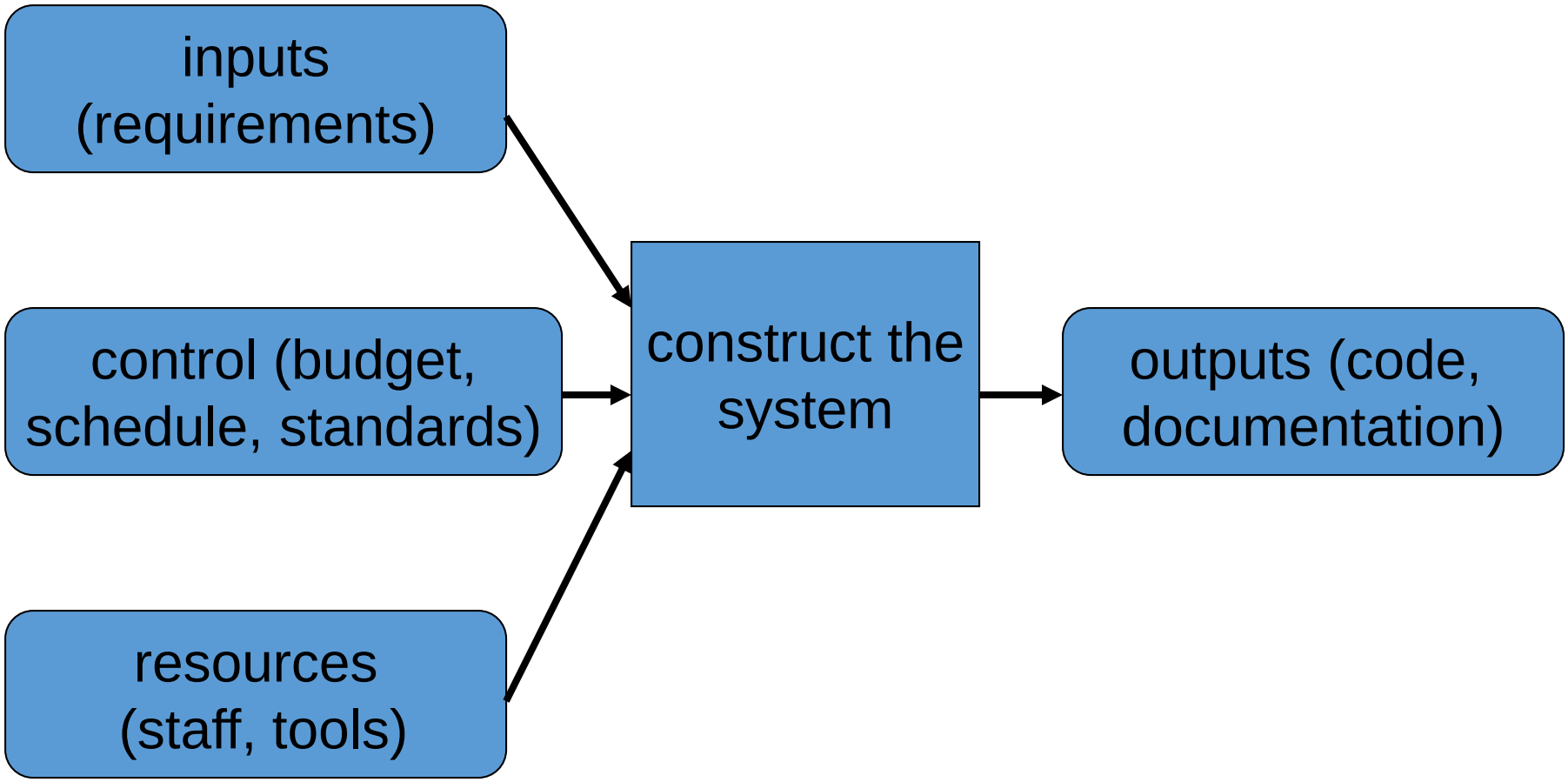
A repeatable model

- The idea is that the process is under ‘statistical control’
- Essentially this means that you are collecting data about the performance of the project

A repeatable model

- This allows you to plan future projects more accurately because, for example, you know approximately how long it will take you to do a new project because you have data on the productivity rates experienced by project teams doing similar work in the past

A repeatable model



Repeatable model KPAs

To move to this level concentrate on:

- Configuration management
  - this is the creation and maintenance of a database of the products being created by the project
  - This enables the project team, among other things, to ensure that when a change is made to one product, e.g. a design document, other related documents, e.g. test cases and expected results are updated

Repeatable model KPAs

To move to this level concentrate on:

- Quality assurance
  - this involves setting up a group whose job it is to check that people are following the laid-down quality procedures

Repeatable model KPAs

To move to this level concentrate on:

- Sub-contract management
- Project planning
- Project tracking and oversight
- Measurement and analysis

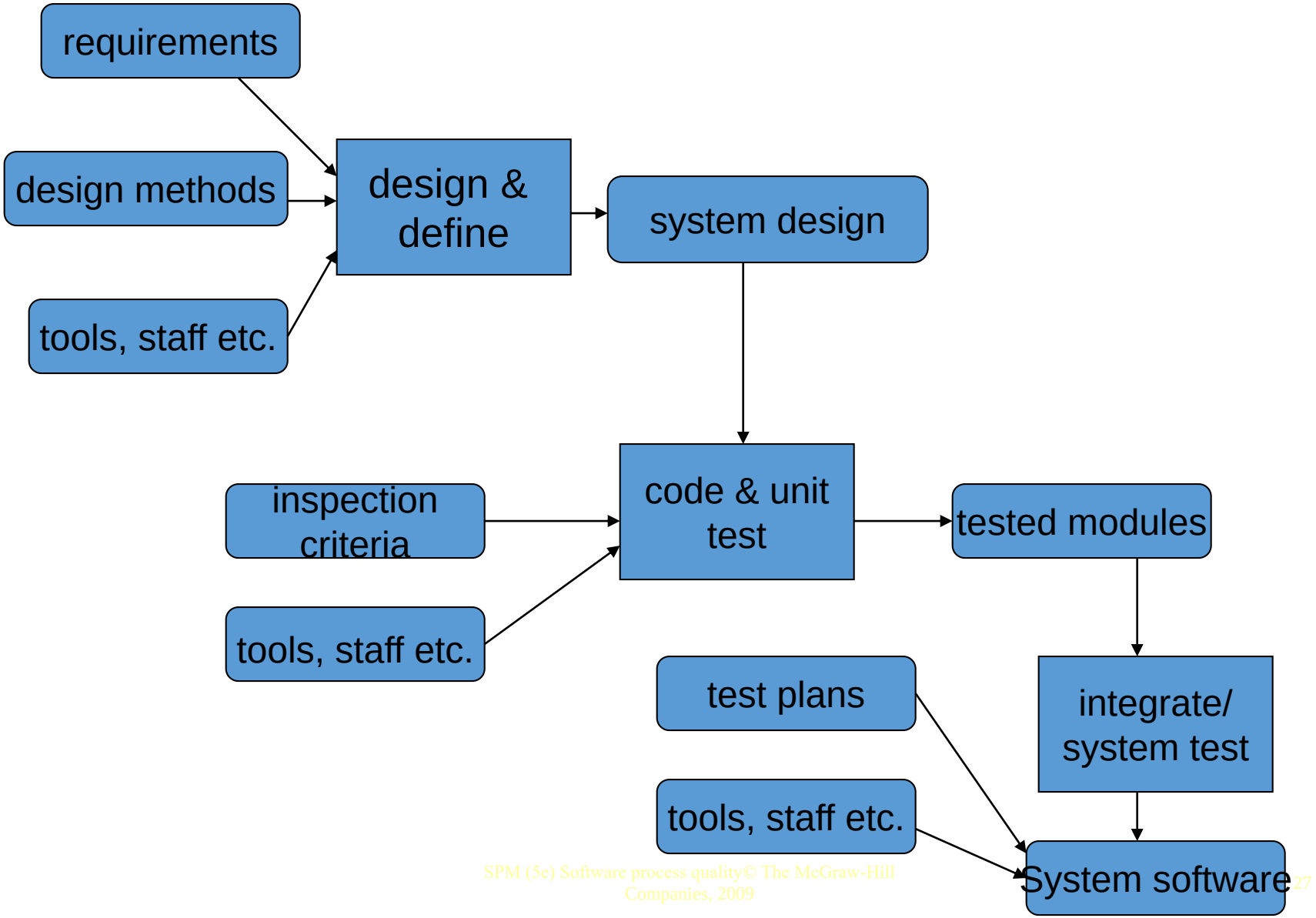
A defined process

- Management at the previous level tends to see the project as a whole
- At this level an attempt is made to breakdown the project into a set of component sub-activities and to make the progress and effectiveness of these sub-activities visible to the wider world

A defined process

- Sub-activities are broken down into even lower level activities until you get to activities carried out by individuals or small teams

A defined process



Repeatable to defined KPAs

Concentrate on

- Requirements development
  - multi-stakeholder requirements evolution
- technical solution
  - evolutionary design and quality engineering

Repeatable to defined KPAs

Concentrate on

- Verification
  - assessment to ensure that the product is produced correctly
- Validation
  - assessment to ensure that the right product is created

Repeatable to defined KPAs

Concentrate on

- Product integration
  - continuous integration, interface control, change management
- Risk management

Repeatable to defined KPAs

Concentrate on

- Organizational training
- Organizational process focus (function)
  - establishing an organizational framework for process definition
- Decision analysis and resolution
  - systematic alternative assessment

Repeatable to defined KPAs

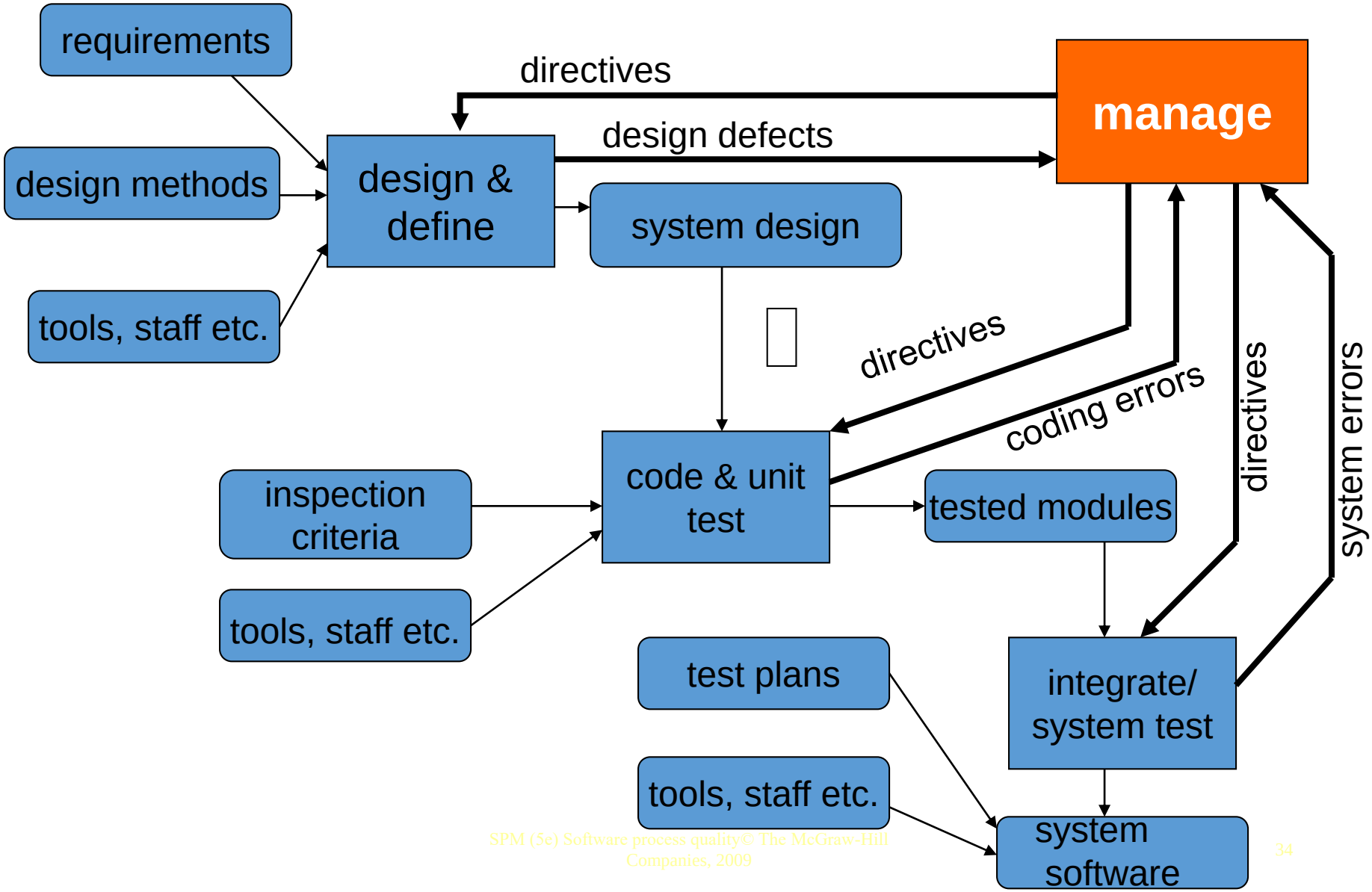
Concentrate on

- Process definition
  - treatment of process as a persistent, evolving asset of the organization
- Integrated project management
  - Methods for unifying the various teams and stakeholders within a project

A managed process

- Now that you have identified individual sub-processes and are collecting data about them, you can carry management interventions as problems start to emerge and thus stop the problems evolving into major set-backs

A Managed Process



Defined to managed KPAs

Concentrate on:

- Organizational process performance
- Quantitative project management

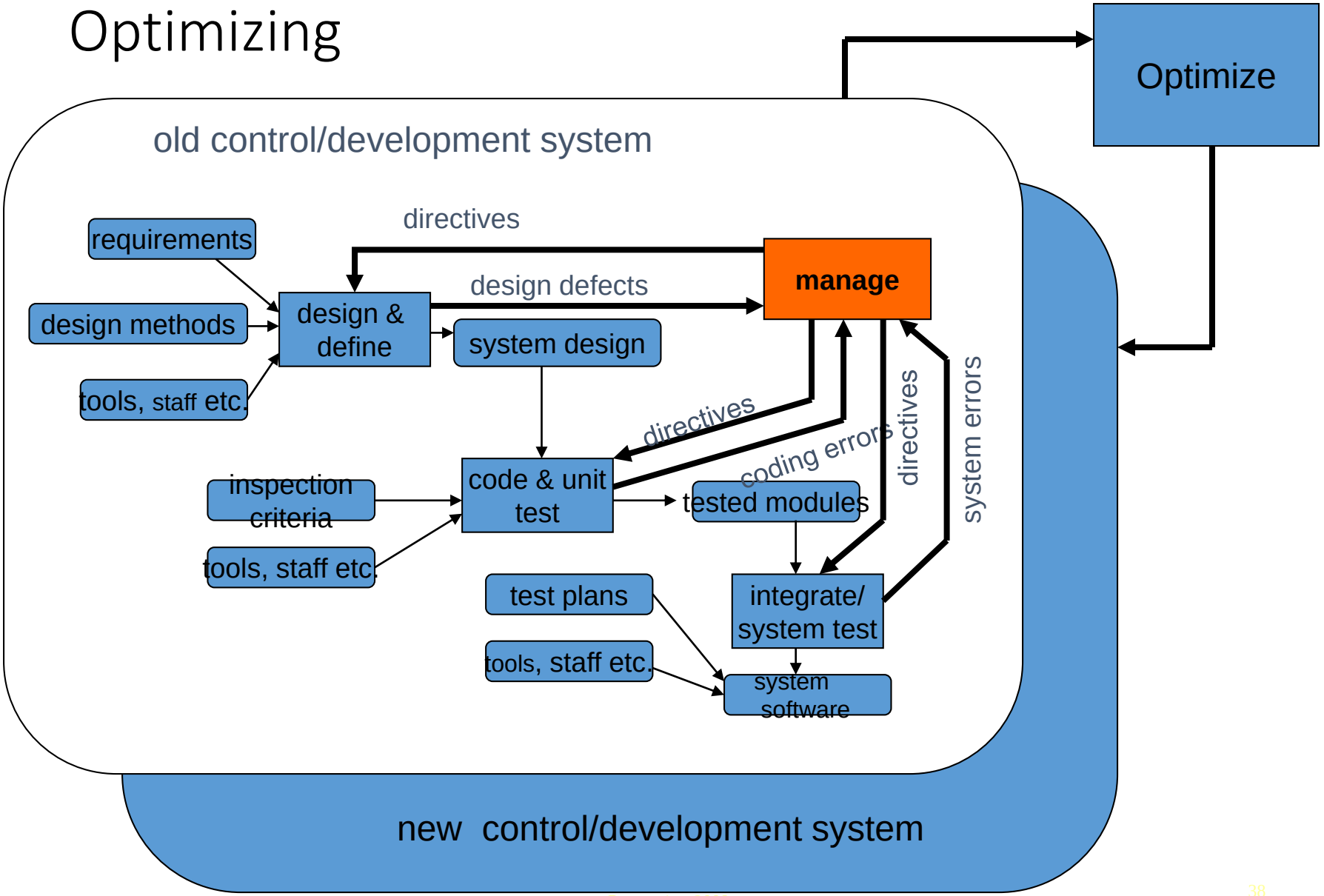
Optimizing

- You are now in a position not only to deal with problems and quality short-falls as they emerge in a project; you can look at the processes you have designed and identify defects in them that lead to deficient products

Optimizing

- Actions can be taken to remove the source of deficient products, not just the deficient products themselves

Optimizing


Optimize
old control/development system
directives
requirements
design methods
tools, staff etc.
design & define
design defects
manage
system design
directives
coding errors
directives
system errors
inspection criteria
tools, staff etc.
code & unit test
tested modules
test plans
tools, staff etc.
integrate/ system test
system software
new control/development system

Managing to optimizing: KPAs

Concentrate on:

- Causal analysis and resolution
- Organizational innovation and deployment

Some questions about CMMI

- Suitable only for large organizations?
  - e.g. need for special quality assurance and process improvement groups
- Defining processes may not be easy with new technology
  - How can we plan when we've not used the development method before?

Some questions about CMMI

- Higher CMM levels easier with maintenance environments?
- Can you jump levels?

ISO/IEC 15504 IT process assessment

- To provide guidance on the assessment of software development processes
- Also known as **SPICE** (Software Process Improvement and Capability Determination)
- It can be argued that CMMI can be seen as a particular implementation of ISO 15504

Process Reference Model

- A defined standard approach to development
- Reflects recognized good practice
- A benchmark against which the processes to be assessed can be judged
- ISO 12207 is the default model
- Could use others in specific environments

ISO 15504 performance attributes for process assessment

CMMI level	ISO 15504
	0. incomplete
initial	1.1.process performance – achieves defined outcome
repeatable	2.1 process management – it is planned and monitored
	2.2 work product management – control of work products

ISO 15504 performance attributes - cont.

CMMI	ISO 15504
Defined	3.1. Process definition
	3.2. Process deployment
Managed	4.1. Process measurement
	4.2. Process control
Optimizing	5.1. Process innovation
	5.2. Process optimization

ISO 15504 Process Assessment

For each process in the relevant Process Reference Model

For each set of attribute level criteria

Assess whether:

N: not achieved 0-15%

P: partially achieved >15%-50%

L: largely achieved >50%-85%

F: fully achieved >85%

ISO 15504 performance indicators

This is just an example of how indicators for each level *might* be identified

1.Performance

Descriptions of maximum and minimum expected input values exist

2.1 Performance management

A plan of how expected input variable ranges are to be obtained exists which is up to date

ISO 15504 performance indicators

2.2 Work product management

There are minutes of a meeting where the input requirements document was reviewed and corrections were mandated

ISO 15504 performance indicators

3.1 Process definition

A written procedure for input requirements gathering exists

3.2 Process deployment

A control document exists that is signed as each part of the procedure is completed

ISO 15504 performance indicators

4.1. Process measurement

Collected measurement data can be collected e.g. number of changes resulting from review

4.2. Process control

Memos relating to management actions taken in the light of the above

ISO 15504 performance indicators

5.1 Process innovation

Existence of some kind of ‘lessons learnt’ report at the end of project

ISO 15504 performance indicators

5.2. Process optimization

Existence of documents assessing the feasibility of suggested process improvements and which show consultation with relevant stakeholders

Techniques to improve software quality

- Increasing visibility
  - "Egoless programming", walk-throughs, inspections and reviews
- Procedural structure
  - Every process has carefully laid down steps

Techniques to improve software quality

- Checking intermediate stages
  - Check the correctness of work at earlier, conceptual, stages

Techniques to improve quality - Inspections

- when a piece of work is completed, copies are distributed to co-workers
- time is spent individually going through the work noting defects

Techniques to improve quality - Inspections

- A meeting is held where the work is then discussed
- A list of defects requiring re-work is produced

Inspections - advantages of approach

- An effective way of removing superficial errors from a piece of software
- Motivates the software developer to produce better structured and self-descriptive code

Inspections - advantages of approach

- Spreads good programming practice
- Enhances team-spirit
- *The main problem* is maintaining the commitment of participants

‘Clean-room’ software development

Ideas associated with Harlan Mills at IBM

- Three separate teams:

1. Specification team – documents user requirements and usage profiles (how much use each function will have)
2. Development team – develops code but does not test it. Uses mathematical verification techniques
3. Certification team – tests code. Statistical model used to decide when to stop

Formal methods

- Use of mathematical notations such as VDM and Z to produce unambiguous specifications
- Can prove correctness of software mathematically (cf. geometric proofs of Pythagoras' theorem)

Formal methods

- Newer approach use Object Constraint Language (OCL) to add detail to UML models
- Aspiration is to be able to generate applications directly from UML+OCL without manual coding – Model Driven Architectures (MDA)

Software quality circles

- Identification of sources of errors
- A group of 4-10 volunteers working in the same are who meet to identify, analyse and solve their work-related problems

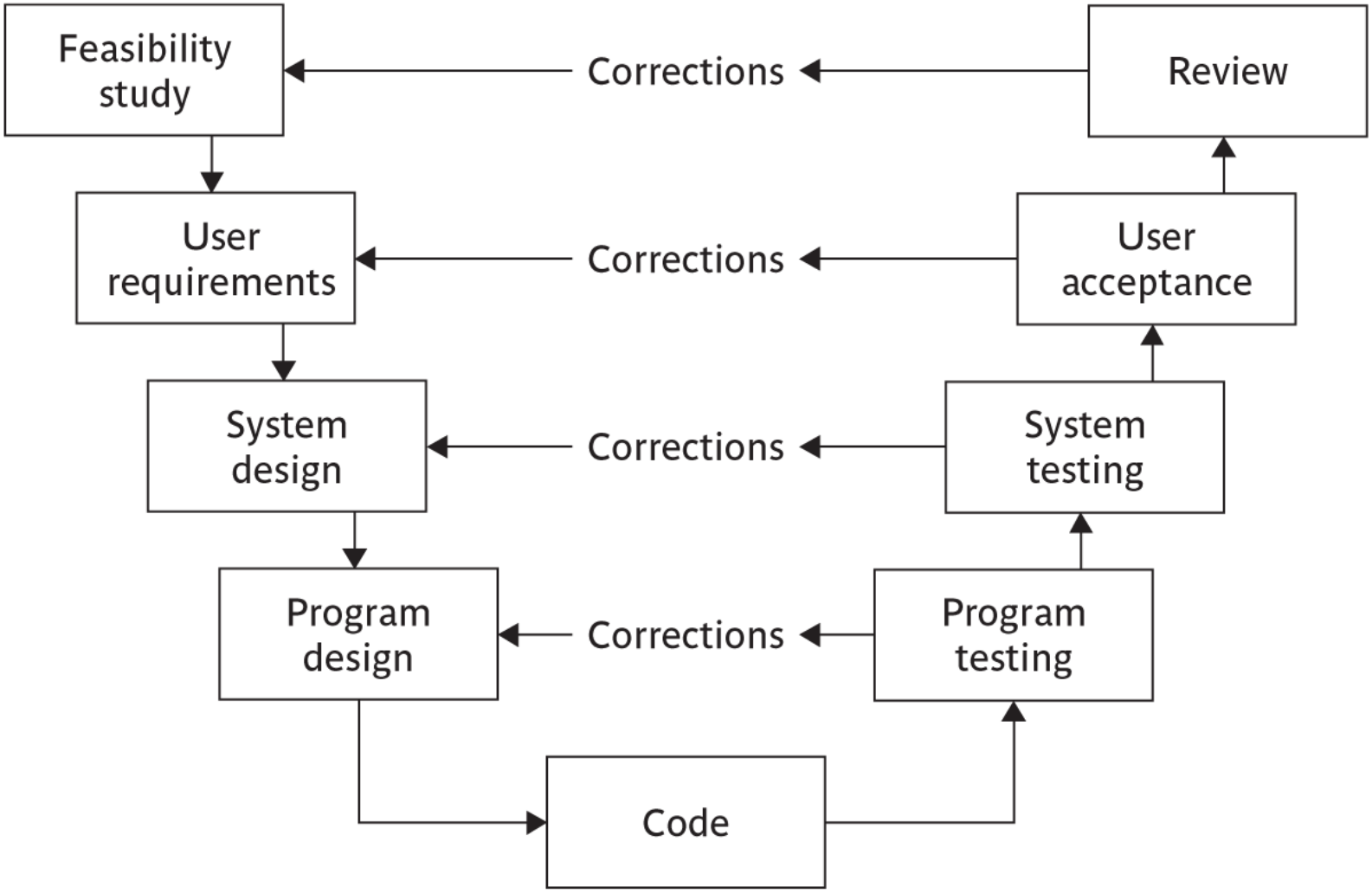
Lessons learnt reports

- Reflecting on the performance of a project at its immediate end
- May identify lessons to be applied to future projects
- PMs are required to write a Lessons Learned report at the end of the project

Testing: the V-process model

- It is an extension of the waterfall approach
- For each development stage there is a testing stage
- The testing associated with different stages serves different purposes e.g. system testing tests that components work together correctly, user acceptance testing that users can use system to carry out their work

Testing: the V-process model



Black box versus glass box test

- Glass box testing
  - The tester is aware of the internal structure of the code; can test each path; can assess percentage test coverage of the tests e.g. proportion of code that has been executed

Black box versus glass box test

- Black box testing
  - The tester is not aware of internal structure; concerned with degree to which it meets user requirements

Test plans

- Specify test environment
  - In many cases, especially with software that controls equipment, a special test system will need to be set up

Test plans

- Usage profile
  - failures in operational system more likely in the more heavily used components
  - Faults in less used parts can lie hidden for a long time
    - E.g. Developer fixes 33-year-old Unix bug
  - Testing heavily used components more thoroughly tends to reduce number of operational failures

Management of testing

The tester executes test cases and may as a result find discrepancies between actual results and expected results – **issues**

Management of testing

Issue resolution – could be:

- a mistake by tester
- a fault – needs correction
- a fault – may decide not to correct: **off-specification**
- a change – software works as specified, but specification wrong: submit to change control

Decision to stop testing

The problem: impossible to know there are no more errors in code

Need to estimate how many errors are likely to be left

Bug seeding – insert (or leave) known bugs in code

Estimate of bugs =

(total errors found)/(seeded errors found) x (total seeded errors)

Bug seeding example

Seed 10 bugs

After the first test, 30 errors are found of which 6 are known errors

Estimate of bugs = 30 / 6 x 10 = 50

We found 30, so 20 errors to be found of which 4 are known

Alternative method of error estimation

- Have two independent testers, A and B
- N_1 = valid errors found by A
- N_2 = valid errors found by B
- N_{12} = number of cases where same error found by A and B
- Estimate = $(N_1 \times N_2)/ N_{12}$
- Example: A finds 30 errors, B finds 20 errors. 15 are common to A and B. How many errors are there likely to be?
- (30x20)/15 = 40

Quality plans

- Quality standards and procedures should be documented in an organization's *quality manual*
- For each separate project, the quality needs should be assessed
- Select the level of quality assurance needed for the project and document in a *quality plan*

Typical contents of a quality plan

- scope of plan
- references to other documents
- quality management, including organization, tasks, and responsibilities
- documentation to be produced
- standards, practices and conventions
- reviews and audits

More contents of a quality plan

- Testing
- Problem reporting and corrective action
- Tools, techniques, and methodologies
- Code, media and supplier control

More contents of a quality plan

- Records collection, maintenance and retention
- Training
- Risk management

Conclusion

- Quality is a vague concept and practical quality requirements have to be carefully defined
- There have to be practical ways of testing for the relative presence and absence of quality

Conclusion

- Most of the qualities that are apparent to the users can only be tested for when the system is completed
- Ways are needed of checking during development what quality of the final system is likely to be