IT/Software Project Management Core Functions

By:
Prof. Dr. Eng. Ghazy Assassa,
CMC-IMC
Certified Management Consultant,
Institute of Management Consultancy, UK

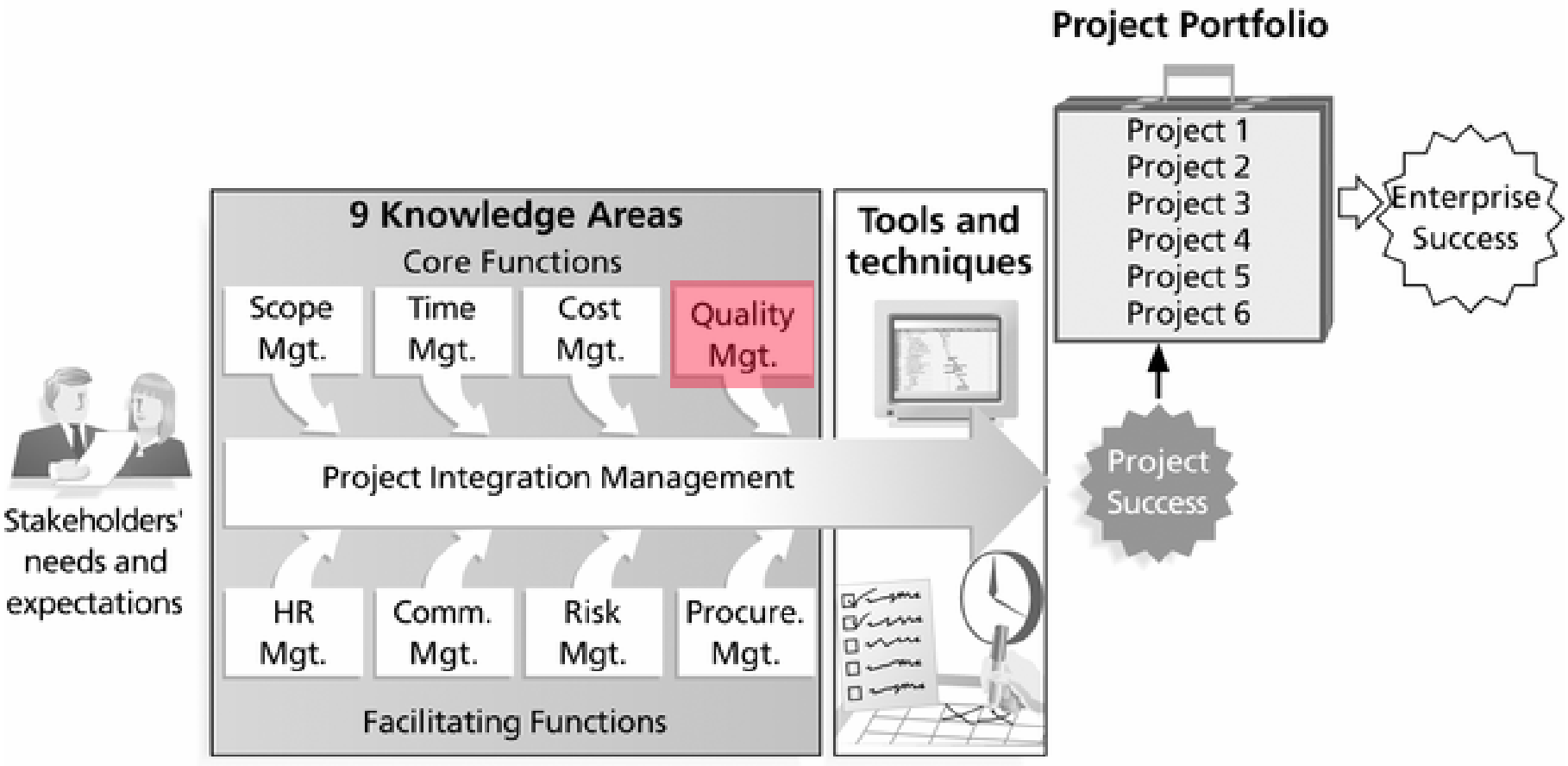
Email: ghazy@ccis.ksu.edu.sa
Mobile: 0502862400

Software Quality Management

Objectives

- To introduce software Quality Management and its relation to project management framework
- To introduce the quality management process and key quality management activities
- To discuss quality models including ISO and CMM
- To explain the role of standards in quality management
- To explain the concept of a software metrics, predictor metrics and control metrics
- To explain how measurement may be used in assessing software quality
- To discuss the concept of reviews in relation to quality management.

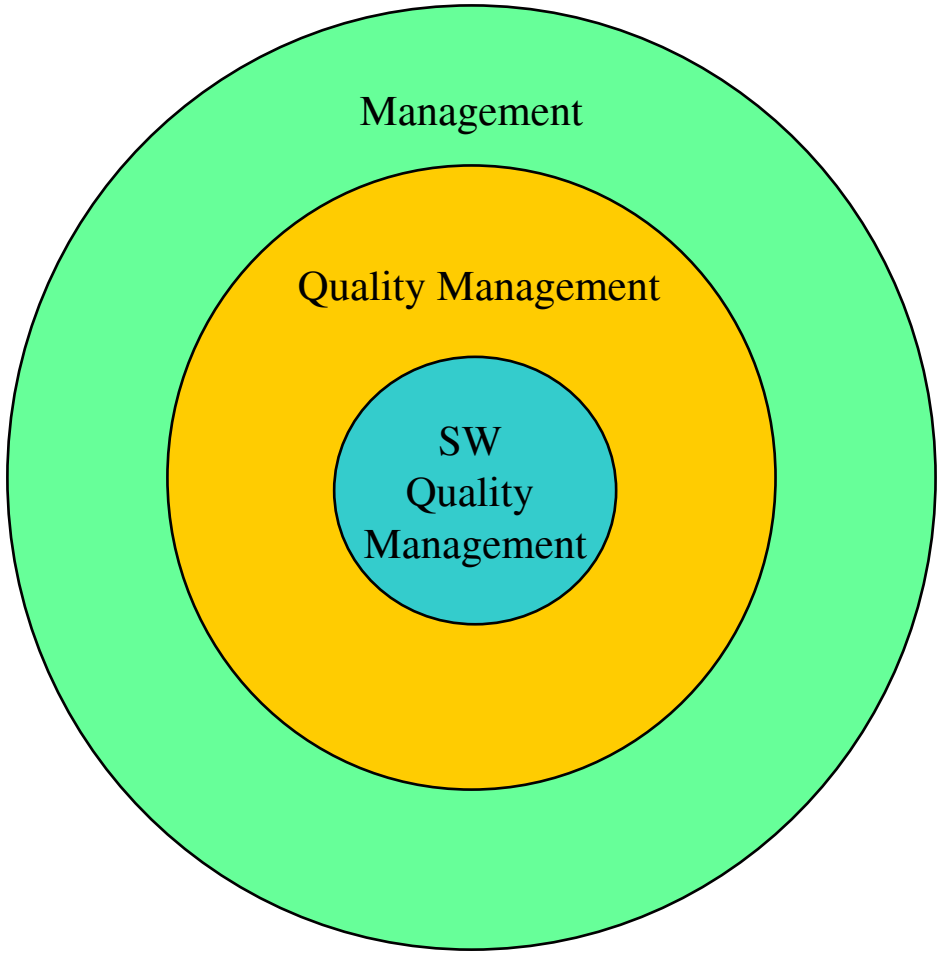
Project Management Framework



IT Project Management, Kathy Schwalbe, Course Technology, 2004.

Quality & The Triple constraint

Management & SW Quality Management



Software Quality Management

- For software industry
- Managing the quality of
  - software process and
  - software products

Quality Concepts

- Quality Assurance (QA) and standards
- Quality planning (QP)
- Quality control (QC)
- Software measurement and metrics

Quality Concepts

- General objective:
  - reduce the “variation between samples” ...
- How does this apply to software?
- Ultimate objective:
  - Customer satisfaction
- Cost of quality
  - appraisal costs
  - failure costs
  - external failure costs

Software Quality Management

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

- Involves
  - defining appropriate quality standards and procedures.
  - and ensuring that these are followed

- Should aim to develop:
  - 'Quality Culture'
  - quality is seen as everyone's responsibility

What is quality?

- Quality: a product should meet its specification

- This is problematical for software systems
  - Tension between (different view)
    - » customer quality requirements (efficiency, reliability, etc.)
    - » and developer quality requirements (maintainability, reusability, etc.)
  - Software specifications are
    - » usually incomplete
    - » and often inconsistent
    - » Solution: RTM (10 quality attributes)

The quality compromise

- We cannot wait for **specifications** to improve before paying attention to quality management
- **Must put** procedures into place to improve quality in spite of **imperfect** specification
- Quality management is therefore not just concerned with reducing defects but also with other product qualities

Quality management activities & levels

- Quality assurance (QA) (organisation level)
- Quality planning (QP) (project level)
- Quality control (QC) (project level)
- Quality review (process and product levels)

Quality management activities (cont.)

- Quality assurance (QA) (organisation level)
  - **Establish** organisational general procedures and standards for quality
  - Analysis, auditing and reporting activities
- Quality planning (QP) (project level)
  - **Select** from QA applicable specific procedures and standards for a particular project and modify these as required
- Quality control (QC) (project level)
  - Ensure that project quality procedures and standards (selected above in QP) are followed (Q review) by the software development team
  - A series of inspections, reviews, tests
- Quality review
  - To validate process and product quality

Quality management activities (cont.)

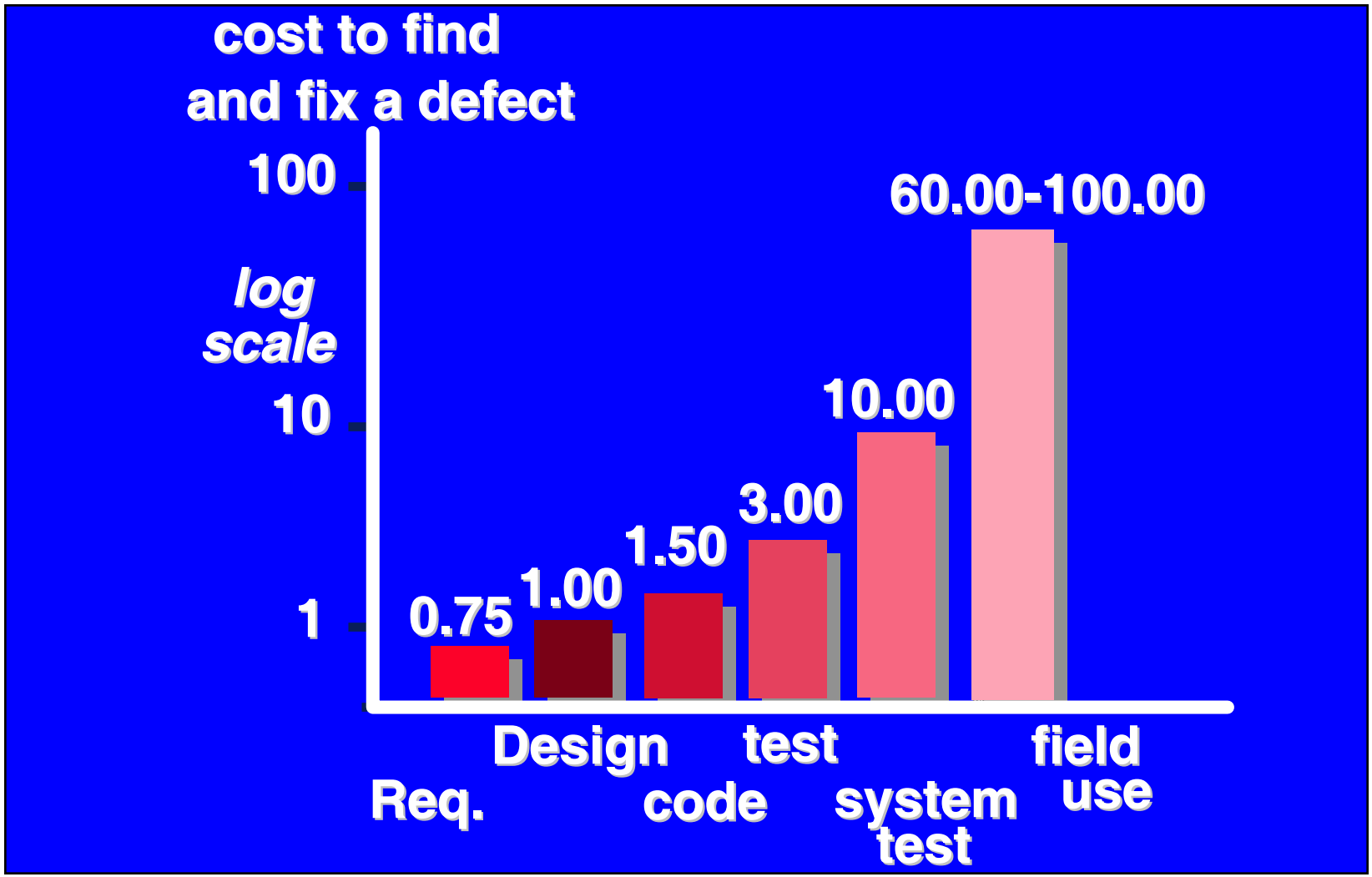
Quality management

should be separated

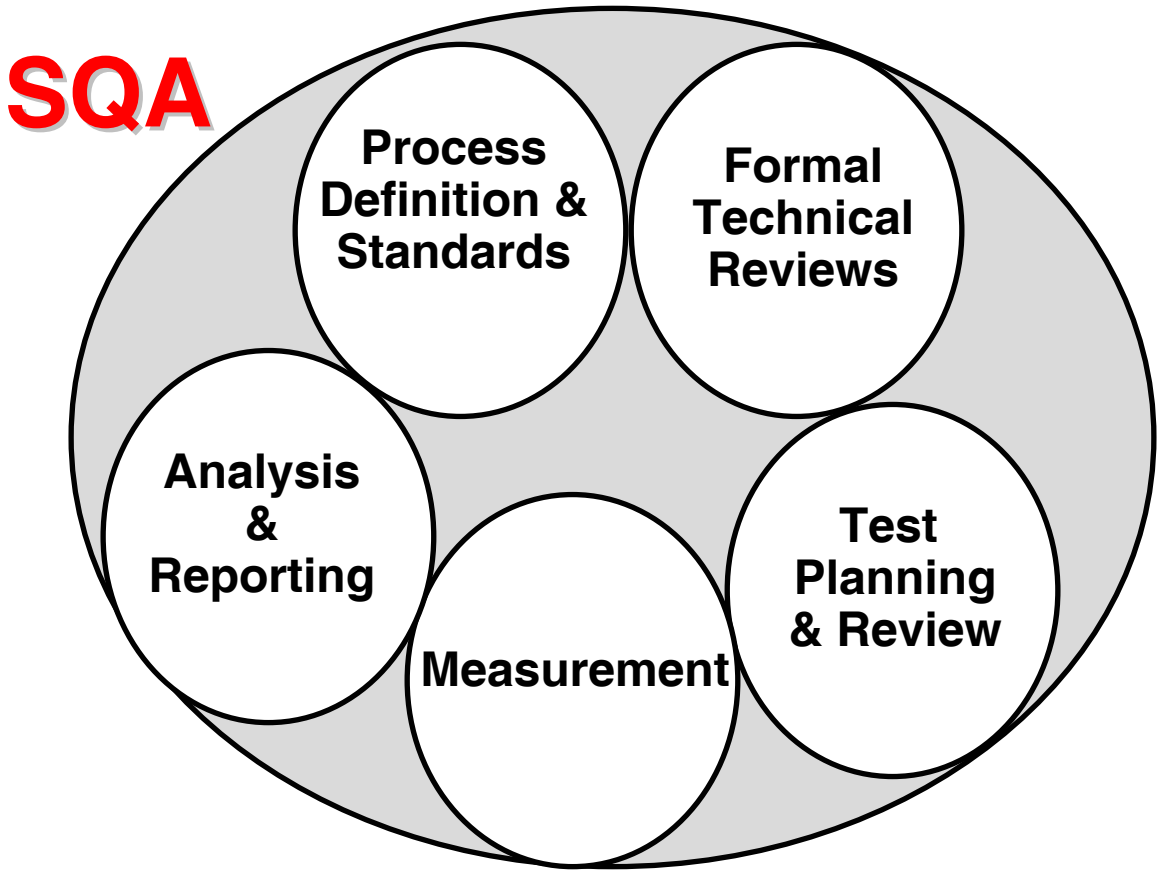
from project management

- Why?
  - To ensure independence
  - To avoid **compromising** quality for project budget or schedule

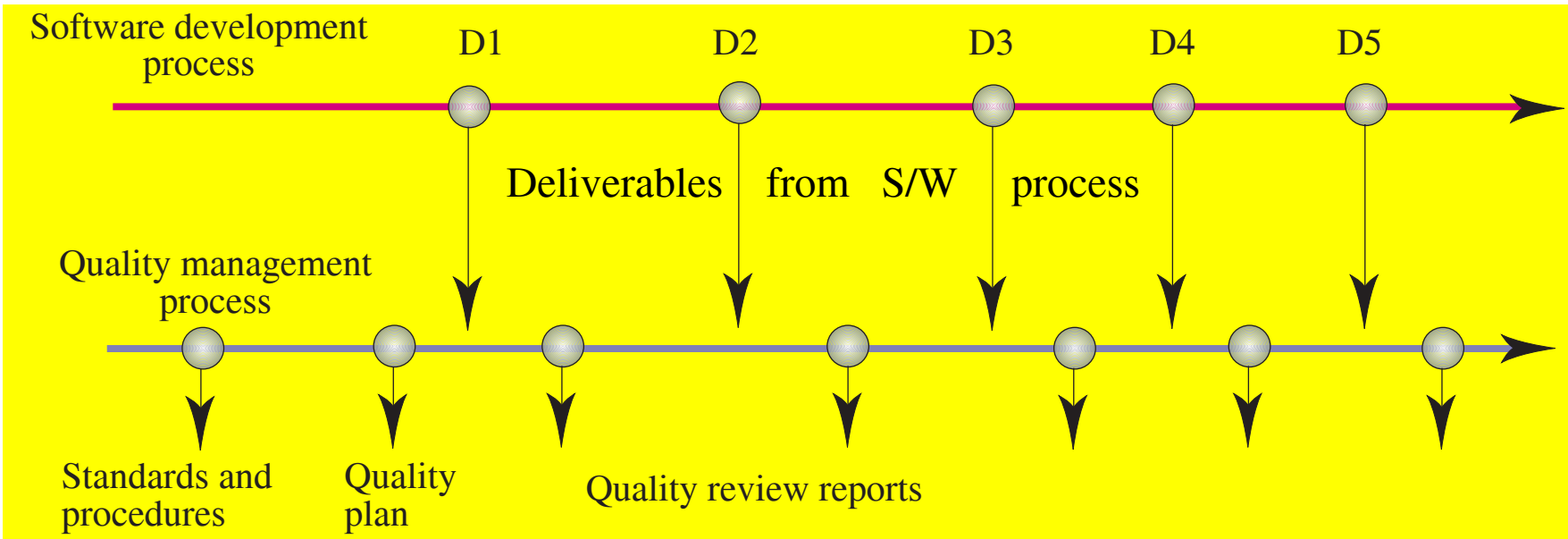
Why SQA Activities Pay Off?



Software Quality Assurance



Quality management and software development



Deliverables from S/W process are **checked** against the defined project standards in the quality control process

Quality Models

- ISO 9000:94 (discontinued - any industry/service)
- ISO 9001:2000 (any industry/service)
- CEMark
- CMM: Capability Maturity Model (software industry)
- TickIT: (software industry)
- ISO 9000-3: (software industry)

Techniques for software process improvements

- Goal Question Metric (GQM)
- Capability Maturity Model (CMM)
- Balanced Scorecard (BSC)
- Six Sigma

Quality Models: ISO, CEMark

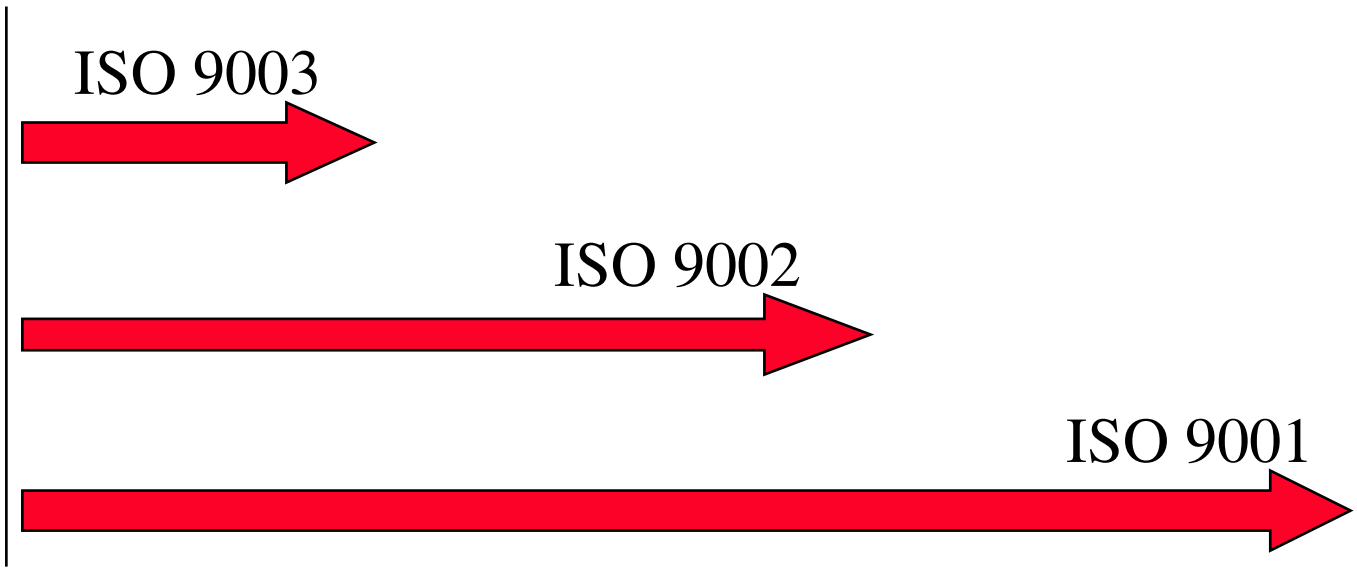
- International set of standards for quality management
- Generic model (Not industry specific) of Quality
  - Applicable to a range of industries from manufacturing to service industries
  - Generic model of quality defining standards & procedures
  - Must be instantiated for each organisation

Software Quality Models: TickIT, CMM

- International set of standards for Software Quality Management (SQM)
- Generic model of software quality defining software standards & procedures
  - Must be instantiated for each software organisation

ISO 9000:94 series (discontinued)

Testing	Product	Product
Maintenance	Manufacturing	Design



ISO 9000:94 Series (discontinued)

- ISO 9001:94
  - applicable to organisations which
    - » Design products
    - » Develop (manufacture) products
    - » and maintain (test) products
  - generic model of the quality process …Must be instantiated for each organisation

Major Changes From ISO 1994 to ISO 2000

- ISO 9002 and 9003 are discontinued
- New architecture (20 clauses become 2+6+4+6+5=23)
- Promotes the adoption of a process approach
- Explicit requirements for customer satisfaction and Continual improvement

- Simplified terminology (easier to use)
  - subcontractor . supplier
  - supplier . organization
  - inspection and testing . product verification and validation
  - quality system element . process
  - quality system . interrelated processes
- For details on ISO see: http://www.sei.cmu.edu/activities/cmm/slides/iso9001cmm.pdf

The ISO 9000 Family of Standards

- Developed to assist organizations, of all types and sizes, to implement and operate effective quality management systems.
- ISO 9000: Quality management systems – Fundamentals and vocabulary
- ISO 9001: Quality management systems – Requirements
- ISO 9004: Quality management systems – Guidelines for performance improvements

ISO 9001

- ISO 9001 process approach –
  - a desired result is achieved more efficiently when activities and related resources are managed as a process.
- Substitute "document" for "say" in the auditor's saying:

  Say what you do; do what you say.

- Document what you do; do what you document.

ISO 9001 & Software: ISO 9000-3

- **ISO 9000-3** is a guideline for applying ISO 9001 to the development, supply, and maintenance of software.

- Originally released in 1991
- Revision to match ISO 9001:2000 has been assigned to ISO/IEC JTC1/SC7 (Software Engineering Standards) subcommittee
- **TickIT** is a U.K. program (British Standards Institute) for applying ISO 9001 to software organizations.
- U.S. and Japanese variants cancelled because of concerns on proliferation of sector-specific standards and interpretation issues

ISO 9001:2000 Quality Model

- What is ISO 9001:2000
  - Set of QMS (Quality Management System) guidelines: Quality model
  - Established by the International Standard Organization

- Who may use it
  - Any organization in
    - » Manufacturing
    - » Service
    - » Distribution
    - » Research & development

  - Any organization in the above sectors to improve its policies, procedures, and documentation system

ISO 9001:2000 Quality Model

- Why using ISO 9001:2000
  - In the global economy, ISO series knows no language/culture barriers
  - An ISO certified company takes significant measures to ensure that the products/services provided are carefully monitored for quality
  - ISO 9000 certification tell customers that the certified company looks for their satisfaction
- When does an organization have to be compliant/certified?
  - Adoption of ISO standards is a voluntary process
  - It takes 9-24 months, depending on organization size and complexity, to adopt ISO standards

ISO 9001

Management responsibility	Quality system
Control of non-conforming products	Design control
Handling, storage, packaging and delivery	Purchasing
Purchaser-supplied products	Product identification and traceability
Process control	Inspection and testing
Inspection and test equipment	Inspection and test status
Contract review	Corrective action
Document control	Quality records
Internal quality audits	Training
Servicing	Statistical techniques

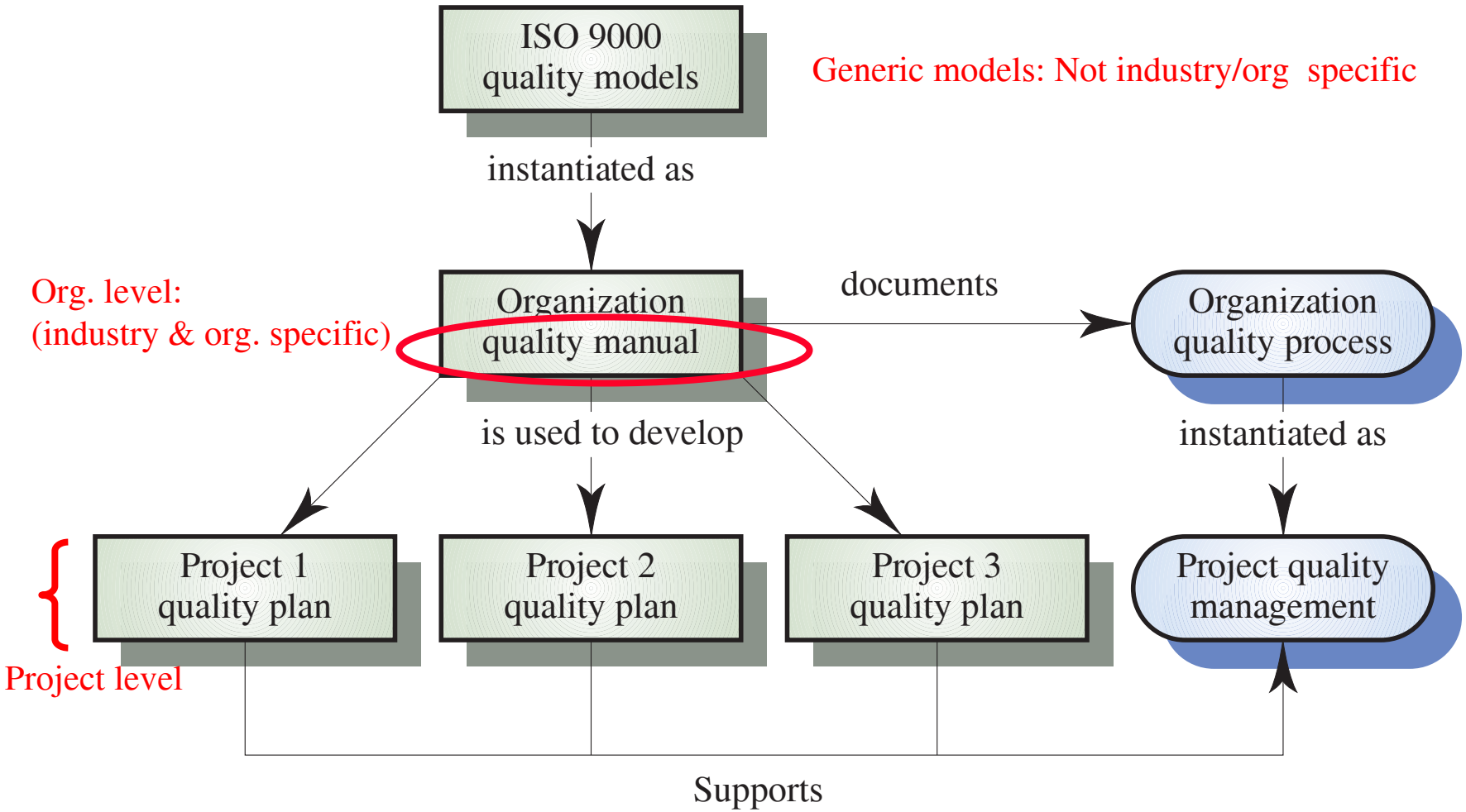
Quality Manual

- Quality standards and procedures followed within the organization
- These quality standards and procedures should be documented in an organisation's Quality Manual
-
- Specific Quality Manual for each organization
- An External body may certify that an organisation's Quality Manual conforms to ISO standards

ISO Certification

- Customers are, increasingly, demanding that suppliers are ISO certified

ISO 9000 and Quality Management



ISO 9000 certification: is / is not

- ISO 9000 certification DOES NOT mean that the quality of the software produced by certified companies will be better than that from uncertified companies.

- ISO 9000 standards are concerned with:
  - the processes to be used in the company, and
  - associated documentations as control processes

- ISO 9000 standards ARE NOT concerned:
  - with ensuring that these processes reflect best practice
  - with product quality

TickIT

- For software industry
- Supported by UK & Swedish software industries
- Started in 1991
- Objectives: Stimulating software developers to think about:
  - What is quality in the context of software development process
  - How software quality may be achieved
  - How Quality Management System may be continuously improved
- TickIT procedures relate directly to the requirements set out in ISO 9001:2000 and to the guidance contained in the Issue 5.0 TickIT Guide

The Capability Maturity Model (CMM) for Software

- A framework (guideline) for software processes improvement
- See Process Improvement chapter (Sommerville)

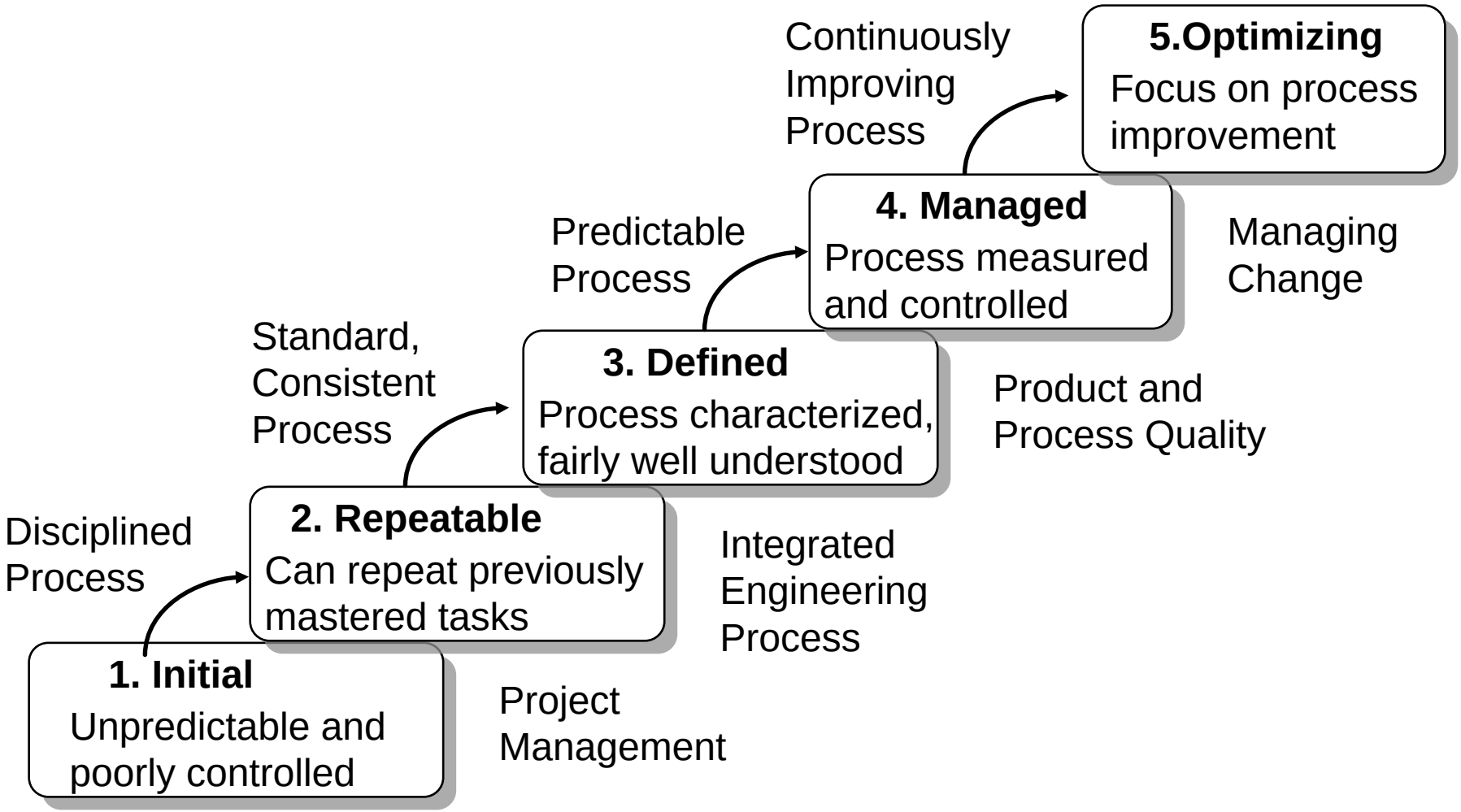
The Capability Maturity Model (CMM) for Software

- Describes the principles and practices underlying software process maturity
- Is intended to help software organizations improve the maturity of their software processes
  - in terms of an evolutionary path from ad hoc, chaotic processes to mature, disciplined software processes.
- CMM is organized into five maturity levels

CMM five maturity levels

1. Initial level (lowest maturity level)
2. Repeatable level
3. Defined level
4. Managed level
5. Optimizing level (highest maturity level)

The Five Levels of Software Process Maturity



CMM

SEI's Vision:

To bring engineering discipline to the development and maintenance of software products

CMM Implementation

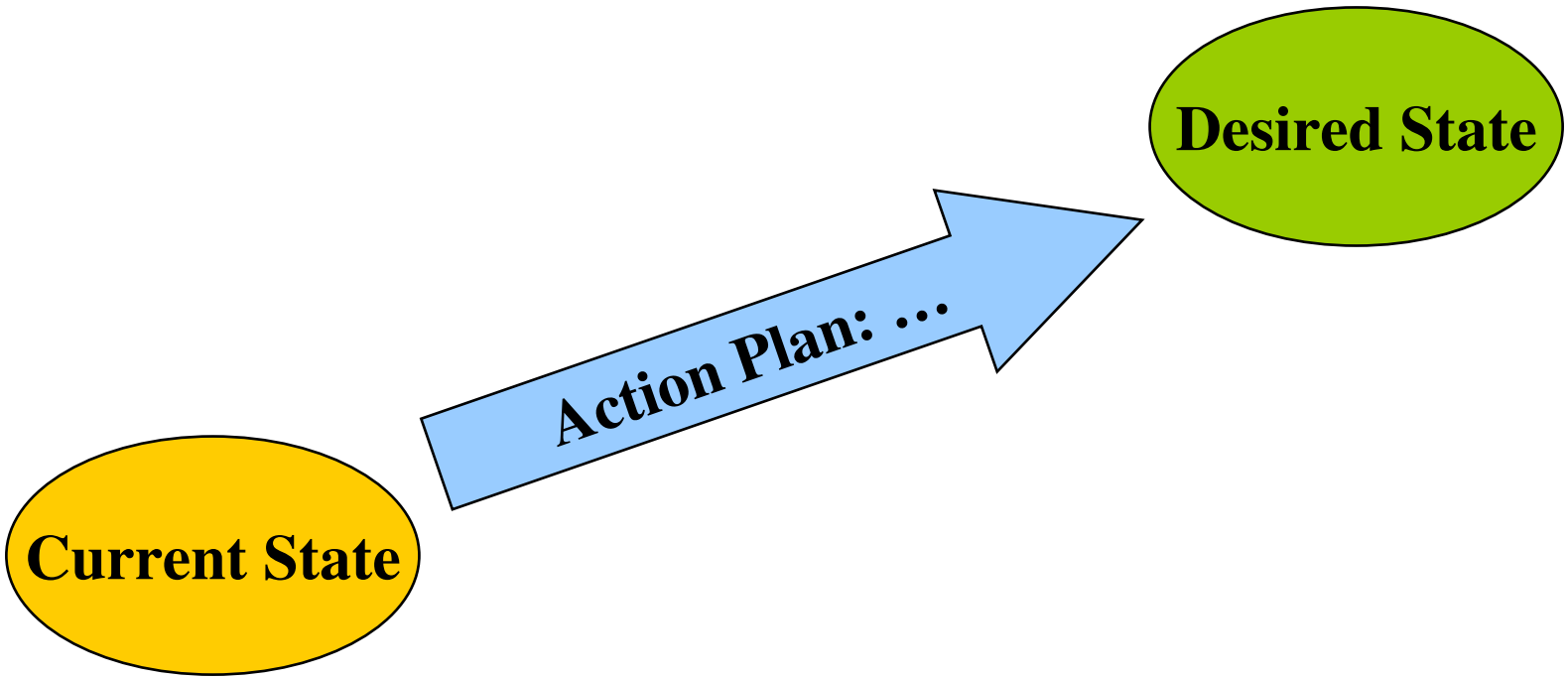
Desired Result:

- **Higher quality**
  - **better products for a better price**
- **Predictability**
  - **function/quality, on time, within budget**

Methodology to Achieve the Desired CMM Result

1. **Identify Current State:**
   Know your current Capability Maturity Level
   Where are you?
2. **Identify Desired State:**
   Understand the description of the next Level
   Where do you want to go?
3. **Act to reduce the Gap:**
   Plan, implement, and institutionalize the key practices of the next Level.
   Repeat until continuous optimization is part of the culture.

Methodology to Achieve the Desired CMM Result



CMM five maturity levels (cont.)

1. Initial level
   - The software process is characterized as ad hoc, and occasionally even chaotic.
   - Few processes are defined, and success depends on individual effort and heroics.

2. Repeatable level
   - Basic project management processes are established to track cost, schedule, and functionality.
   - The necessary process discipline is in place to repeat earlier successes on projects with similar applications.

CMM five maturity levels (cont.)

3. Defined level
   - The software process for both management and SW engineering activities is documented, standardized, and integrated into a standard software process for the organization.
   - All projects use an approved, tailored version of the organization's QM standard software process for developing and maintaining software.
4. Managed level
   - Detailed measures of the software process and product quality are collected.
   - Both the software process and products are quantitatively understood and controlled
5. Optimizing level
   - Continuous process improvement is enabled by quantitative feedback from the process and from piloting innovative ideas and technologies.

CMM five maturity levels: key process areas

- Each maturity level is decomposed into several key process areas that indicate the areas an organization should focus on to improve its software process.
- **Level 1 (Initial):** No decomposition
- **Level 2 (Repeatable):** Focus on the software project's concerns related to establishing basic project management controls.
  - Requirements Management,
  - Software Project Planning,
  - Software Project Tracking and Oversight,
  - Software Subcontract Management,
  - Software Quality Assurance, and
  - Software Configuration Management

CMM five maturity levels: key process areas (cont.)

- **Level 3 (Defined):** Address both project and organizational issues, as the organization establishes an infrastructure that institutionalizes effective software engineering and management processes across all projects.
  - Organization Process Focus,
  - Organization Process Definition,
  - Training Program,
  - Integrated Software Management,
  - Software Product Engineering,
  - Intergroup Coordination, and
  - Peer Reviews

CMM five maturity levels: key process areas (cont.)

- **Level 4 (Managed):** Focus on establishing a quantitative (via measurements) understanding of both the software process and the software work products being built.
  - Quantitative Process Management (via measurements) and
  - Software Quality Management.
- **Level 5 (Optimizing):** Cover the issues that both the organization and the projects must address to implement continual, measurable software process improvement.
  - Defect Prevention,
  - Technology Change Management,
  - and Process Change Management.

CMM five maturity levels: key process areas (cont.)

- For a more detailed overview of the CMM, see:
  - Mark C. Paulk, Bill Curtis, Mary Beth Chrissis, and Charles V. Weber, "Capability Maturity Model, Version 1.1," IEEE Software, Vol. 10, No. 4, July 1993, pp. 18-27.
- or the CMM itself. Version 1.1 of the CMM, which was released in 1993, is now available as a book:
  - Carnegie Mellon University, Software Engineering Institute (Principal Contributors and Editors: Mark C. Paulk, Charles V. Weber, Bill Curtis, and Mary Beth Chrissis), *The Capability Maturity Model: Guidelines for Improving the Software Process*, ISBN 0-201-54664-7, Addison-Wesley Publishing Company, Reading, MA, 1995.

CMM & ISO Models

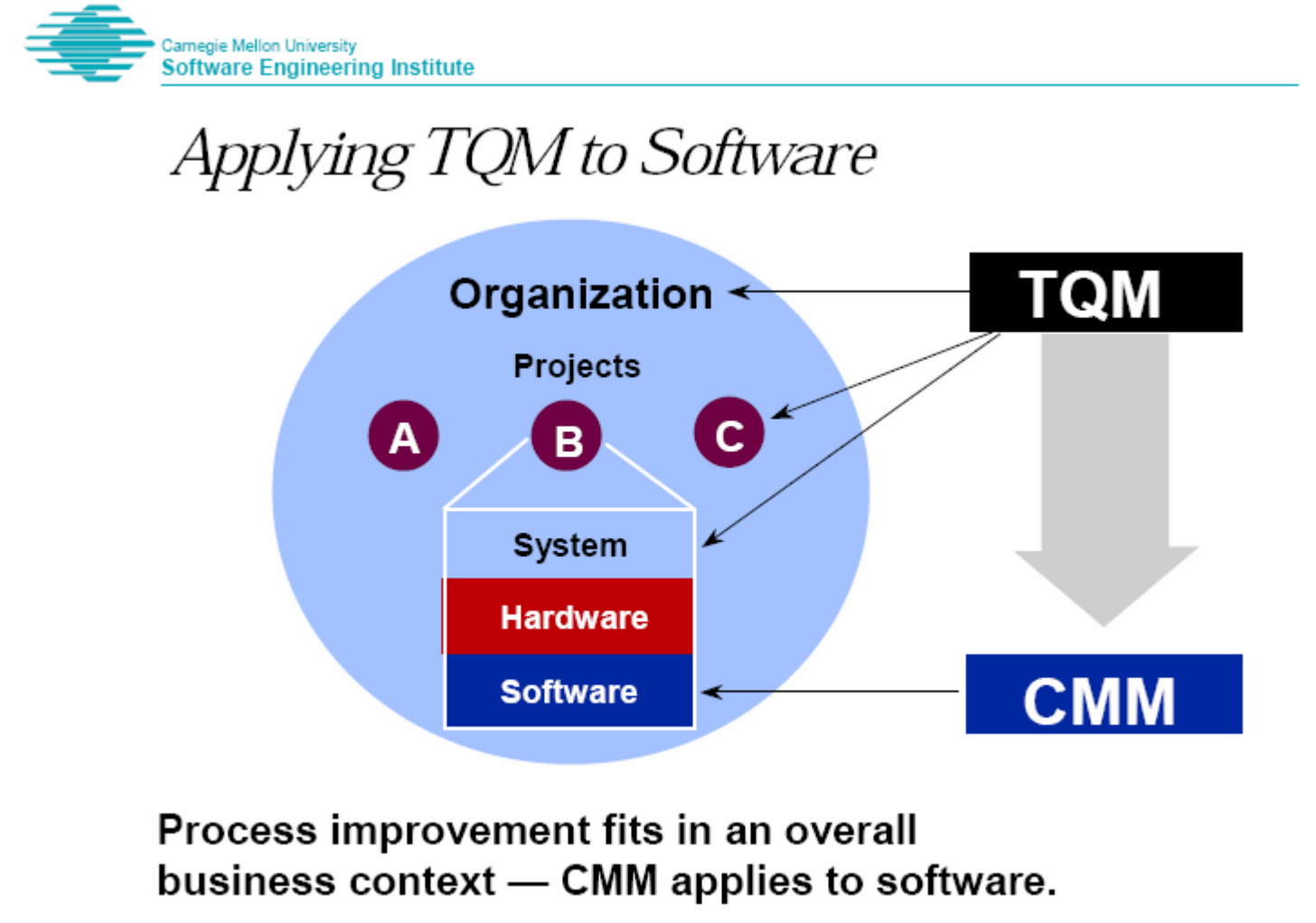
The Software Engineering Institute (SEI) at Carnegie Mellon

- Federally funded research and development center (FFRDC) established in 1984
- Mission–to provide leadership in advancing the state of the practice of software engineering to improve the quality of systems that depend on software
- Vision–to bring engineering discipline to the development and maintenance of software

The Capability Maturity Model for Software (SW-CMM)

- A common-sense application of the concepts of Total Quality Management to software projects
- A five-level prescriptive model for organizational transformation
- A community-developed guide with descriptions of good engineering and management practices
- The basis for reliable and consistent CMM-based appraisals

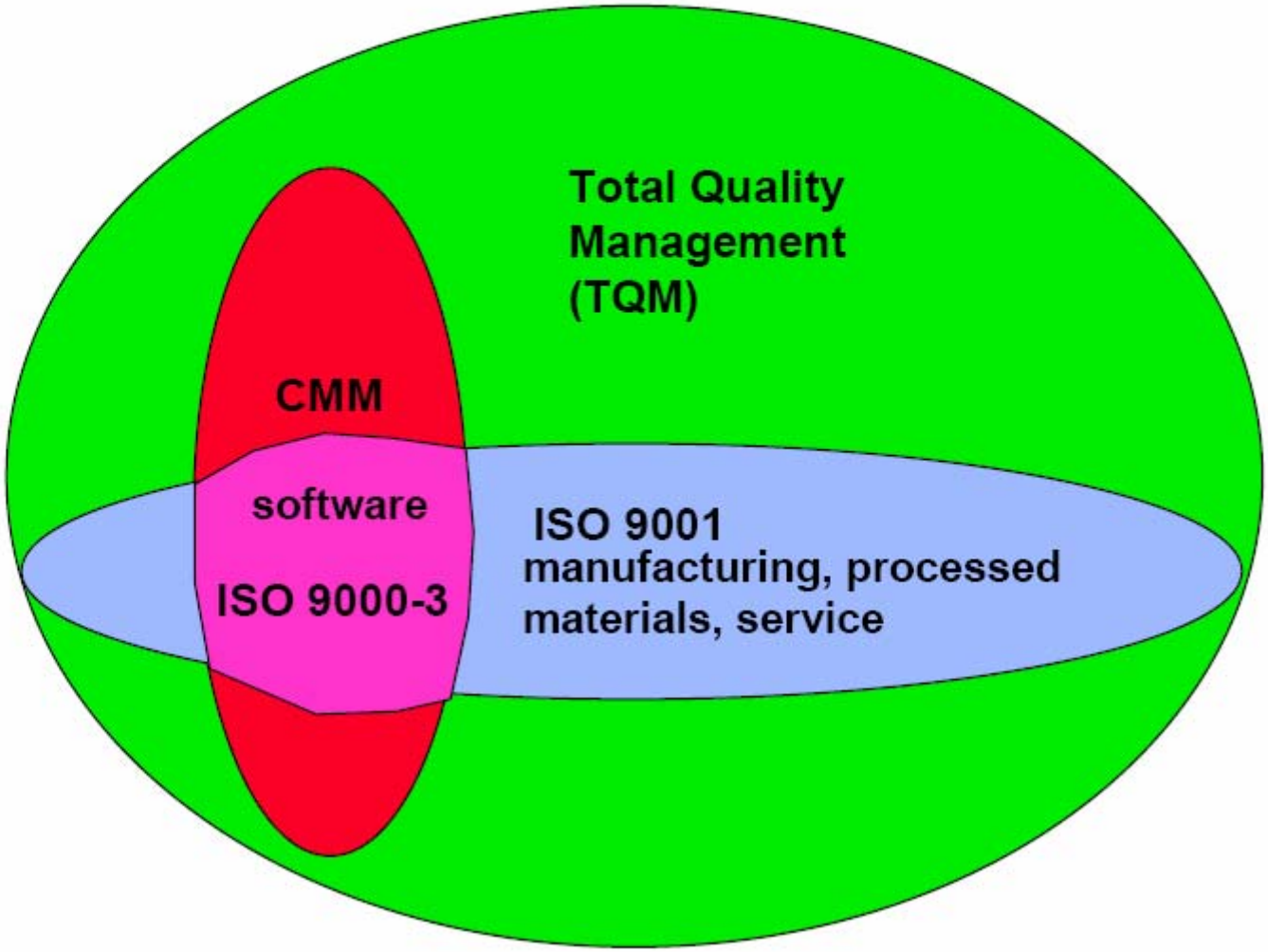
Applying TQM to Software



Software CMM

Level	Focus	Key Process Areas
5 Optimizing	*Continual process improvement*	Defect Prevention Technology Change Management Process Change Management
4 Managed	*Product and process quality*	Quantitative Process Management Software Quality Management
3 Defined	*Engineering processes and organizational support*	Organization Process Focus Organization Process Definition Training Program Integrated Software Management Software Product Engineering Intergroup Coordination Peer Reviews
2 Repeatable	*Project management processes*	Requirements Management Software Project Planning Software Project Tracking & Oversight Software Subcontract Management Software Quality Assurance Software Configuration Management
1 Initial	*Competent people (and heroics)*	

TQM, CMM, and ISO 9001



ISO Program

- Should **not** be thought of as a process improvement program.
- Bias towards **documenting ..** and thus standardizing **existing processes** rather than improving.
- May create a bias to change..
  - It is best to implement ISO after fundamental improvements have been made.

Common Themes Between ISO 9001 and Software CMM

- Emphasis on process
- Documented processes
- Practiced processes
- What, not how
- Shared principles

Process Management Premise

- The quality of a (software) system is highly influenced by the **quality of the process** used to develop and maintain it.
- Process is not the only success factor –you also need
  - competent people
  - appropriate business models and strategic plans
  - product innovation
  - integration across design and production (breaking down organizational barriers)

Documented Processes

- In the Software CMM, processes are documented:
  - procedures, standards, and methods
  - organizational policies
- ISO 9001 process approach –a desired result is achieved more efficiently when activities and related resources are managed as a process.
- Substitute "document" for "say" in the auditor's proverb: "Say what you do; do what you say".

- **"Document** what you do; *show me that you* do what you **document"**

What, Not How

- Software CMM and ISO 9001 describe **what a process should address** rather than how the process should be implemented.
  - Software CMM is over 400 pages long
  - Heart of ISO 9001 is about 12 pages long
- Organizations have to define their own processes.
  - Processes can be described with the minimum essential information –concise, elegant, useful, usable, used.
  - Process (or quality) manuals can be 8**00-page** monstrosities – baroque, complex, unusable, not useful (and not used).

Mapping ISO 9001 to Software CMM

Level 2		*Level 3*		*Level 4*	
RM	√ √	OPF	√ √	QPM	√
SPP	√	OPD	√ √	SQM	√ √
SPTO	√	TP	√ √		
SSM	√ √	ISM	√	*Level 5*	
SQA	√ √	SPE	√ √	DP	√
SCM	√ √	IC	√	TCM	
		PR	√	PCM	√

√ partially addressed by ISO 9001

√ √ largely (not completely?) addressed by ISO 9001

(perhaps by inference, judgment was used)

(in the proper environment)

The Level of an ISO 9001 Compliant Organization

- What level in the Software CMM would an ISO 9001 compliant organization be at?
- In principle, should be at least a strong Level 2 organization and probably Level 3
  - Possibly at Level 1, if the focus is strictly on getting the ISO 9001 certificate: many cases where a Level 1 organization has gotten ISO 9001 certification

Quality assurance and standards

- Standards are the key to effective quality management

- Standards may be
  - international,
  - national,
  - organizational or
  - project standards (per type of project)

Quality Assurance
Product and process standards

- Quality Assurance includes setting up quality standards for product and process
- Product standards
  - Define what characteristics all components should exhibit e.g. a common programming style
  - Includes
    - » Requirements documentation standards
    - » Software documentation standards
- Process standards
  - Define how the software process should be enacted
  - Includes:
    - » Specification process standards
    - » Design process standards
    - » Implementation (coding) process standards
    - » Testing process standards

Product and process standards

What

How the software process should be enacted

Product standards	Process standards
Design review form	Design review conduct
Document naming standards	Submission of documents to CM
Procedure header format	Version release process
Ada programming style standard	Project plan approval process
Project plan format	Change control process
Change request form	Test recording process

Importance of standards

- Encapsulation of best practice- avoids repetition of past mistakes
- Framework for quality assurance process - it involves checking standard compliance
- Provide continuity - new staff can understand the organisation by understanding the standards applied

Bodies developing standards

- National & International bodies
- IEEE
- ANSI
- BS
- NATO
- US DoD

- Defense organisation (US DoD, NATO) may require that their own standards are followed in software contracts

Problems with standards

- Not seen as relevant and up-to-date by software engineers
- Involve too much bureaucratic form filling
- Unsupported by software tools .. so tedious manual work is involved to maintain 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 associated tool support. Excessive clerical work is the most significant complaint against standards

Process and product quality

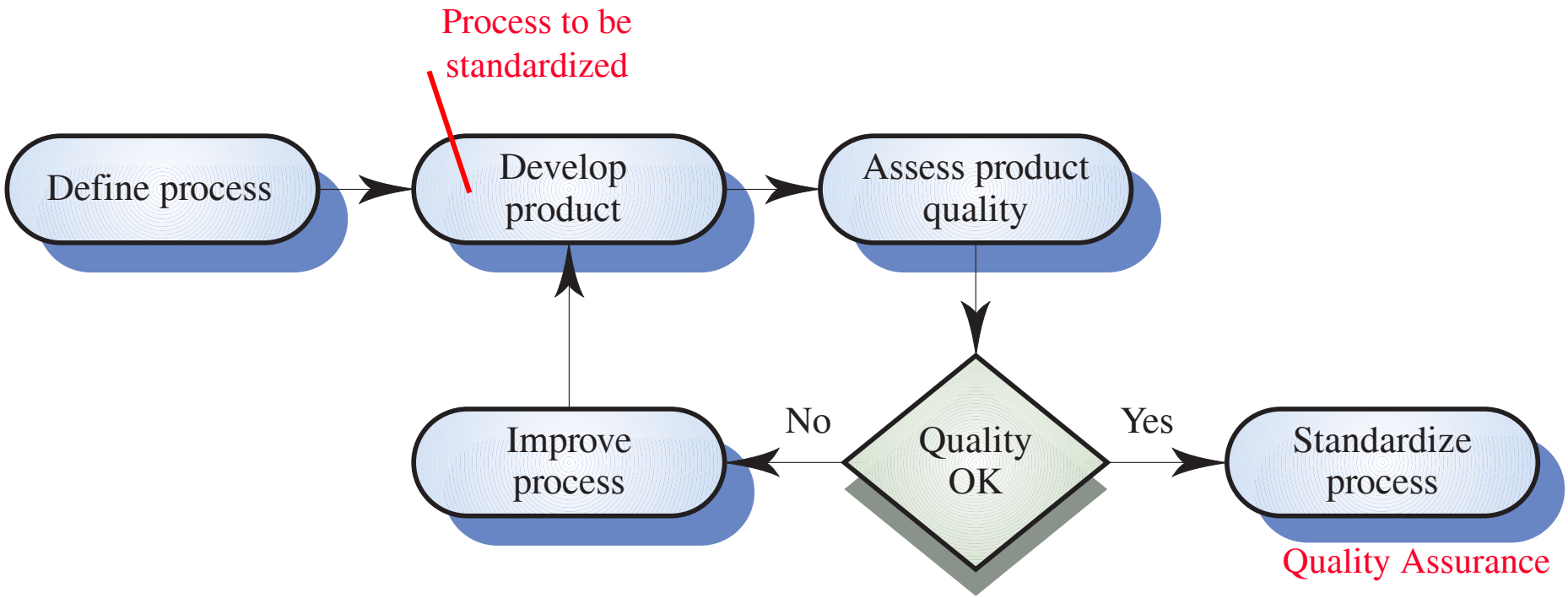
- Process quality has a significant influence on the quality of the software

- there is a very complex relationship between software processes and product quality

Process-based quality

- Manufactured goods
  - Straightforward link between process and product because the process is relatively easy to standardise and monitor
  - Once manufacturing systems are calibrated, they can be run again and again to produce high quality products
- More complex for software because:
  - Software is not manufactured but designed
  - Software development is a creative rather than mechanical process
  - External factors such as the novelty of an application or the need for an accelerated development schedule may affect product quality irrespective of the process used
- Care must be taken not to impose inappropriate process standards

QA & Process Improvement - Process-based quality



Practical process quality

- Define process standards such as
  - how reviews should be conducted,
  - how configuration management should be implemented, etc.
- Monitor the development process to ensure that standards are being followed
- Report on the process to project management and software procurer

Quality planning

- A quality plan sets out
  - the desired product qualities and how these are assessed
  - and define the most significant quality attributes
- It should define the quality assessment process, an agreed upon what ‘high quality’ software actually means
- It should set out which organisational standards (specified in the Quality Manual) should be applied and, if necessary, define new standards

Quality plan structure

- Product introduction:
  - Product description, its intended market, quality expectations
- Product plans
  - Critical release dates, distribution plans, product servicing
- Process descriptions
  - Development & service processes to be used in development & management
- Product quality goals
  - Including identification & justification of critical product quality attributes and how to achieve them
- Quality assessment process
  - How product quality will be checked
- Risks and risk management
  - For key risks that might affect product quality

Quality plan structure (cont.)

- Quality plans should be short, succinct documents
  - If they are too long, no-one will read them

Software quality attributes

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

Part of Quality Planning is identification of critical quality attributes and planning how to achieve them

Quality Control (QC)

- Checking the software development process to ensure that QA procedures and standards are being followed

- Two approaches to quality control
  - Quality reviews
  - Automated software assessment and software measurement

- What Are Reviews?

What Are Reviews?

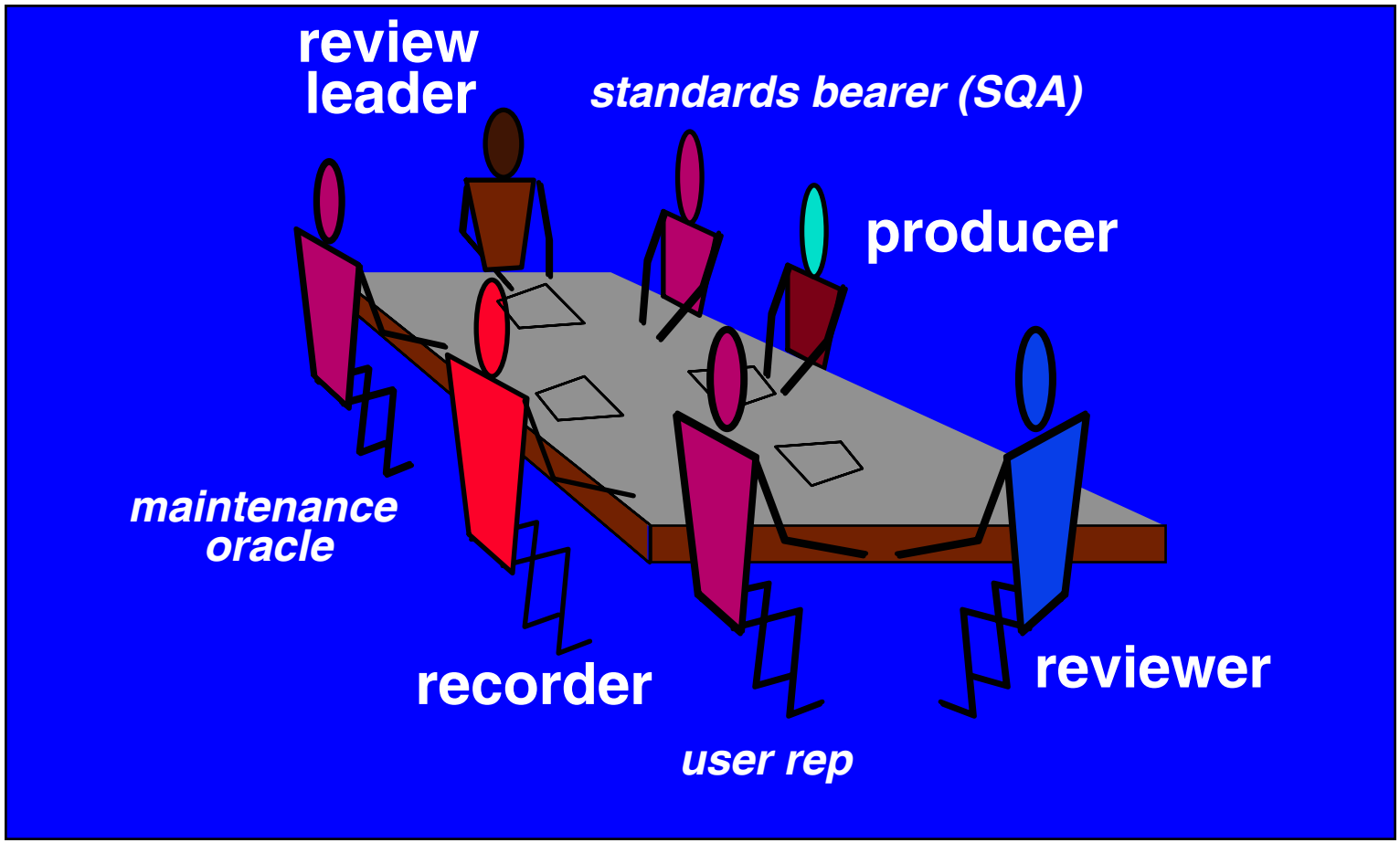
- a meeting conducted by technical people for technical people
- a technical assessment of a work product created during the software engineering process
- a software quality assurance mechanism
- a training ground

What Reviews Are Not!

They are not:

- a project budget summary
- a scheduling assessment
- an overall progress report
- a mechanism for reprisal or political intrigue!!

Reviews Players



Metrics Derived from Reviews

- inspection time per page of documentation
- inspection time per KLOC or FP
- inspection effort per KLOC or FP
- errors uncovered per reviewer hour
- errors uncovered per preparation hour
- errors uncovered per SE task (e.g., design)
- number of minor errors (e.g., typos)
- number of major errors
  (e.g., nonconformance to req.)
- number of errors found during preparation

Quality Reviews

- The principal method of validating the quality of a process or of a product
- A group examine, part of or all, S/W documentations & processes used to produce it and check that project standards are followed and that S/W & documents conform to these standards
- Deviations are reported to project manager

Types of review

Review type	Principal purpose
Design or program inspections	To detect detailed errors in the design or code and to check whether standards have been followed. The review should be driven by a checklist of possible errors.
Progress reviews	To provide information for management about the overall progress of the project. This is both a process and a product review and is concerned with costs, plans and schedules.
Quality reviews	To carry out a technical analysis of product components or documentation to find faults or mismatches between the specification and the design, code or documentation. It may also be concerned with broader quality issues such as adherence to standards and other quality attributes.

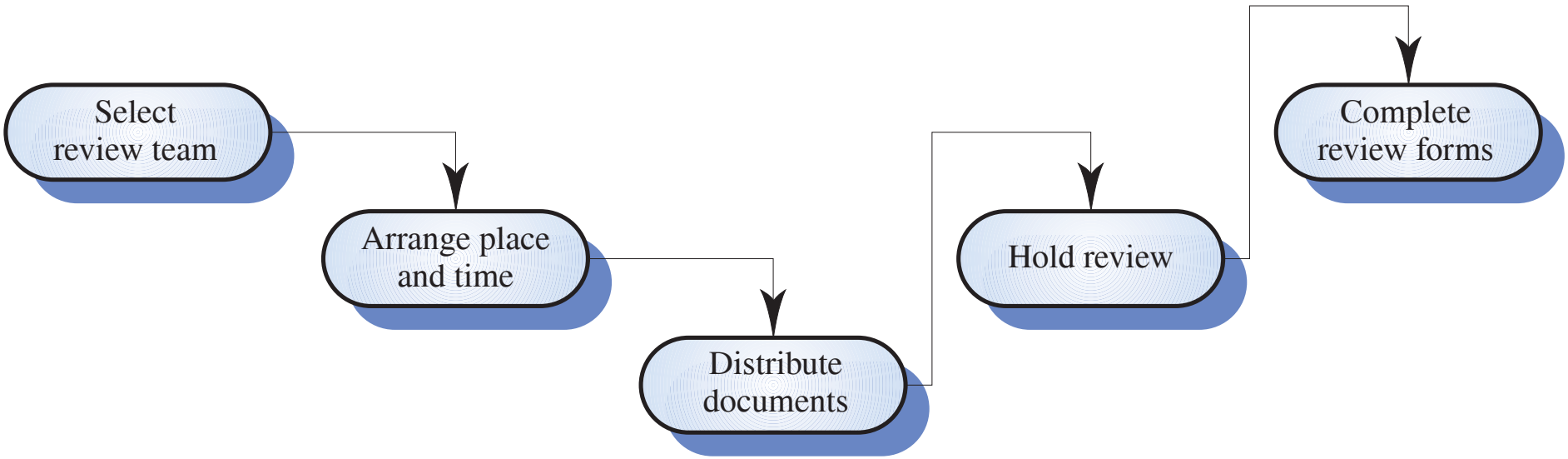
Quality reviews

- A group of people carefully examine part or all of a software system and its associated documentation.
- Reviews are document-based but are not limited to Code, designs, specifications
- Documents such as test plans, process model, process standards, user manuals, etc. may be reviewed.
- Documents to be reviewed must be distributed in advance to reviewers
- The review should be short (2 hours max)
- The author of the document being reviewed should walk through the document with the review team

Quality reviews

- 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 review process



Review functions

- Quality function
  - they are part of the general quality management process
- Project management function
  - they provide information for project managers
- Training and communication function
  - product knowledge is passed between development team members

Quality reviews

- Objective is the discovery of system defects and inconsistencies
- Any documents produced in the process may be reviewed
- Review teams should be relatively small and reviews should be fairly short
- Review should be recorded and records maintained

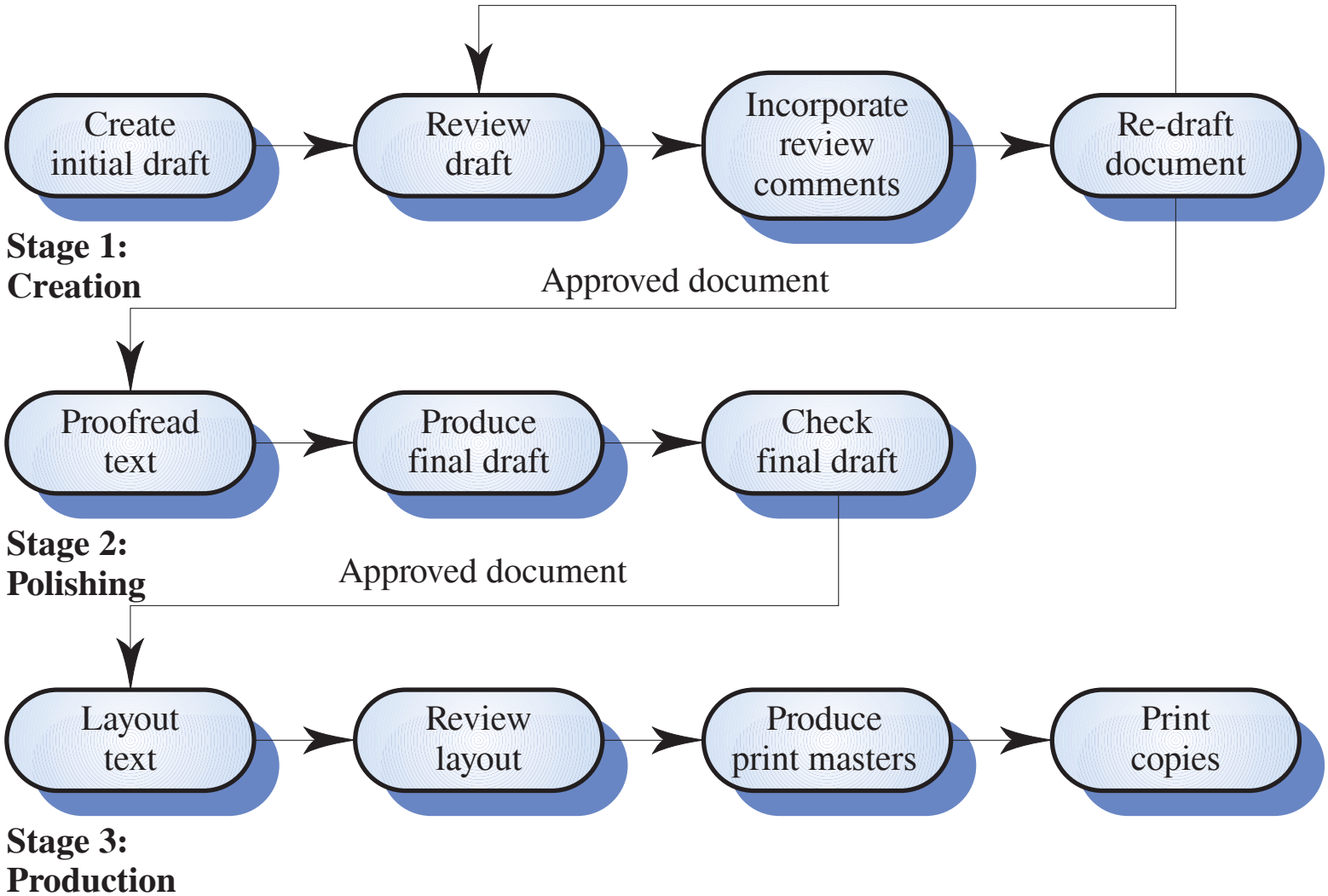
Review results

- Comments made during the review should be classified.
  - No action. No change to the software or documentation is required.
  - Refer for repair. Designer or programmer should correct an identified fault.
  - Reconsider overall design. The problem identified in the review impacts other parts of the design. Some overall judgement must be made about the most cost-effective way of solving the problem.
- Requirements and specification errors may have to be referred to the client.

Documentation standards

- Documents are the tangible manifestation of the software
- Documentation process standards
  - How documents should be developed, validated and maintained
- Document contents standards
  - Contents, structure, and appearance
- Document interchange standards
  - How documents are stored and interchanged between different documentation systems

Documentation process



Document contents standards

- Document identification standards
  - How documents are uniquely identified
- Document structure standards
  - Standard structure for project documents
- Document presentation standards
  - Define fonts and styles, use of logos, etc.
- Document update standards
  - Define how changes from previous versions are reflected in a document

Document interchange standards

- Documents are produced using different systems and on different computers
- Interchange standards allow electronic documents to be exchanged, mailed, etc.
- Need for archiving. The lifetime of word processing systems may be much less than the lifetime of the software being documented
- XML as an emerging standard for document interchange

Software measurement and metrics

- Software measurement is concerned with deriving a numeric value for an attribute of a software product or process
- This allows for objective comparisons between techniques and processes
- Although some companies have introduced measurement programmes, the systematic use of measurement is still uncommon
- There are few standards in this area

Software metrics

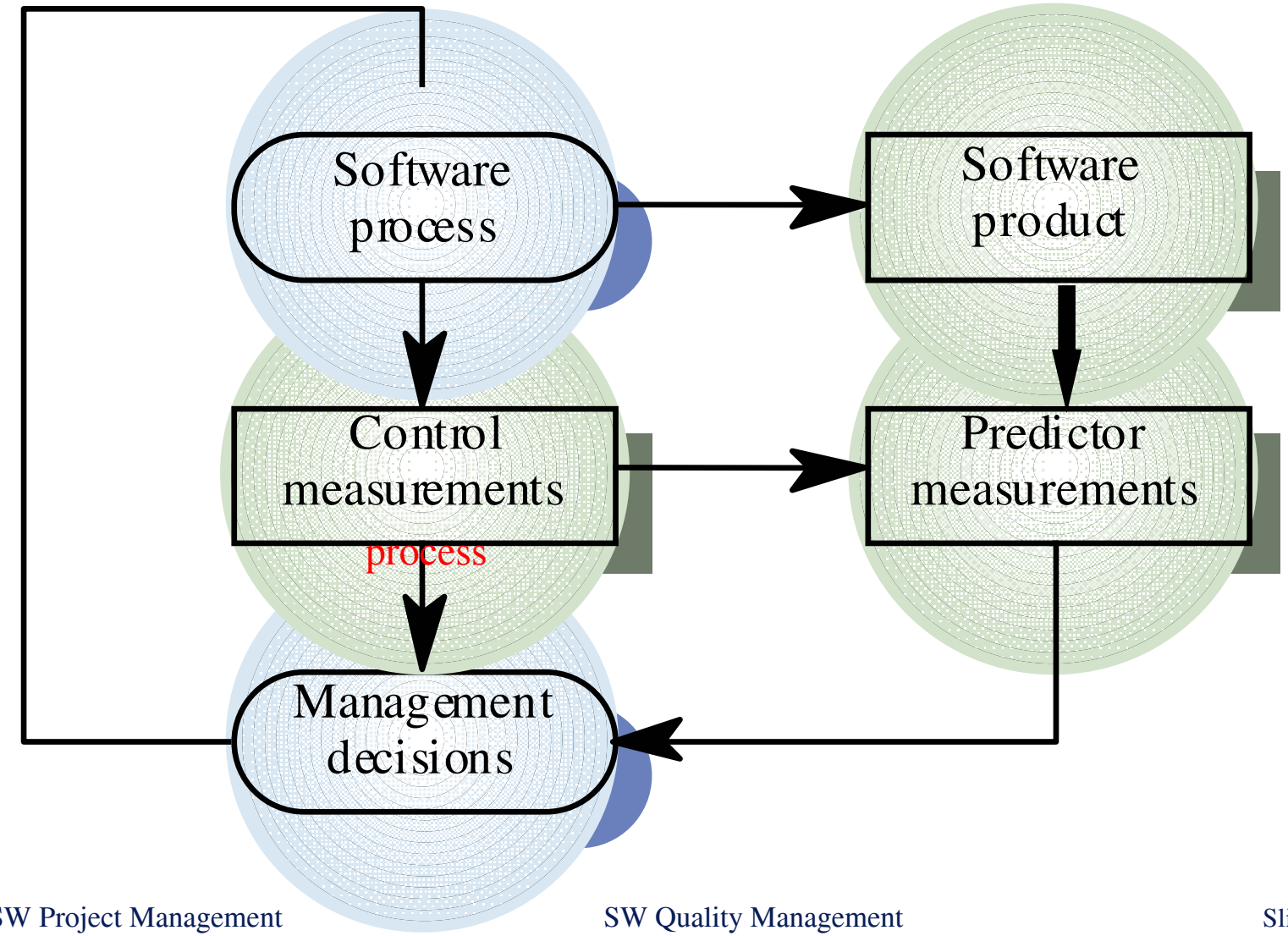
- Any type of measurement which relates to a software system, process or related documentation
  - Lines of code in a program, the Fog index ‘readability of a passage of written text’, number of person-days required to develop a component,..
- Allow the software and the software process to be quantified
- Measures of the software process or product

Software metrics

- Software metrics may be
  - Predictor metrics : predict product attributes
  - Control /process metrics: to control the software process

- Predictor metrics example:
  - Cyclomatic complexity of a module
  - Number of attributes and methods within class deign
- Control /process metrics example:
  - Average effort and time to repair a defect

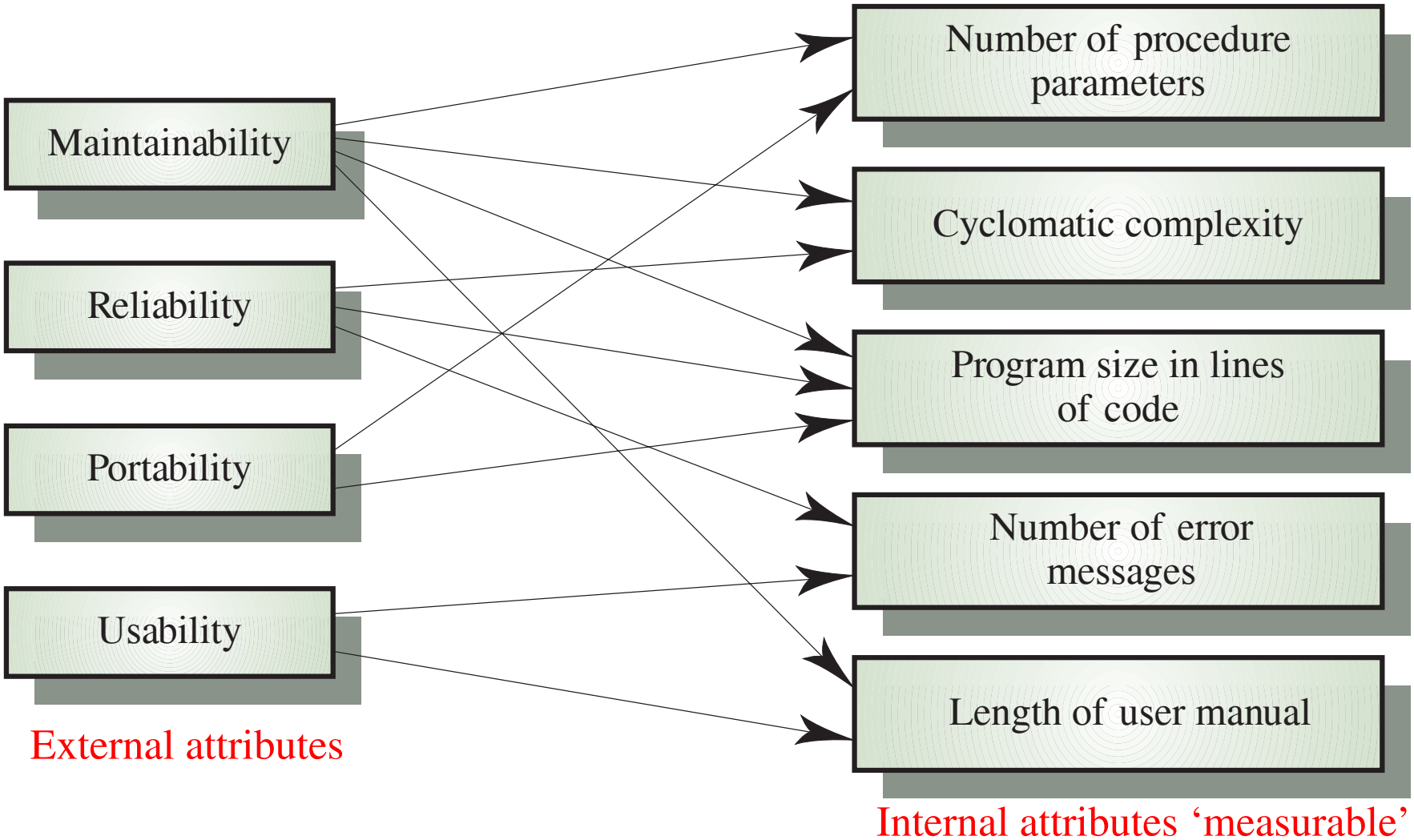
Predictor and control metrics



Metrics assumptions

- A software property can be measured
- The relationship exists between what we can measure and what we want to know
- This relationship has been formalized and validated
- It may be difficult to relate what can be measured to desirable quality attributes (maintainability, complexity, understandably)

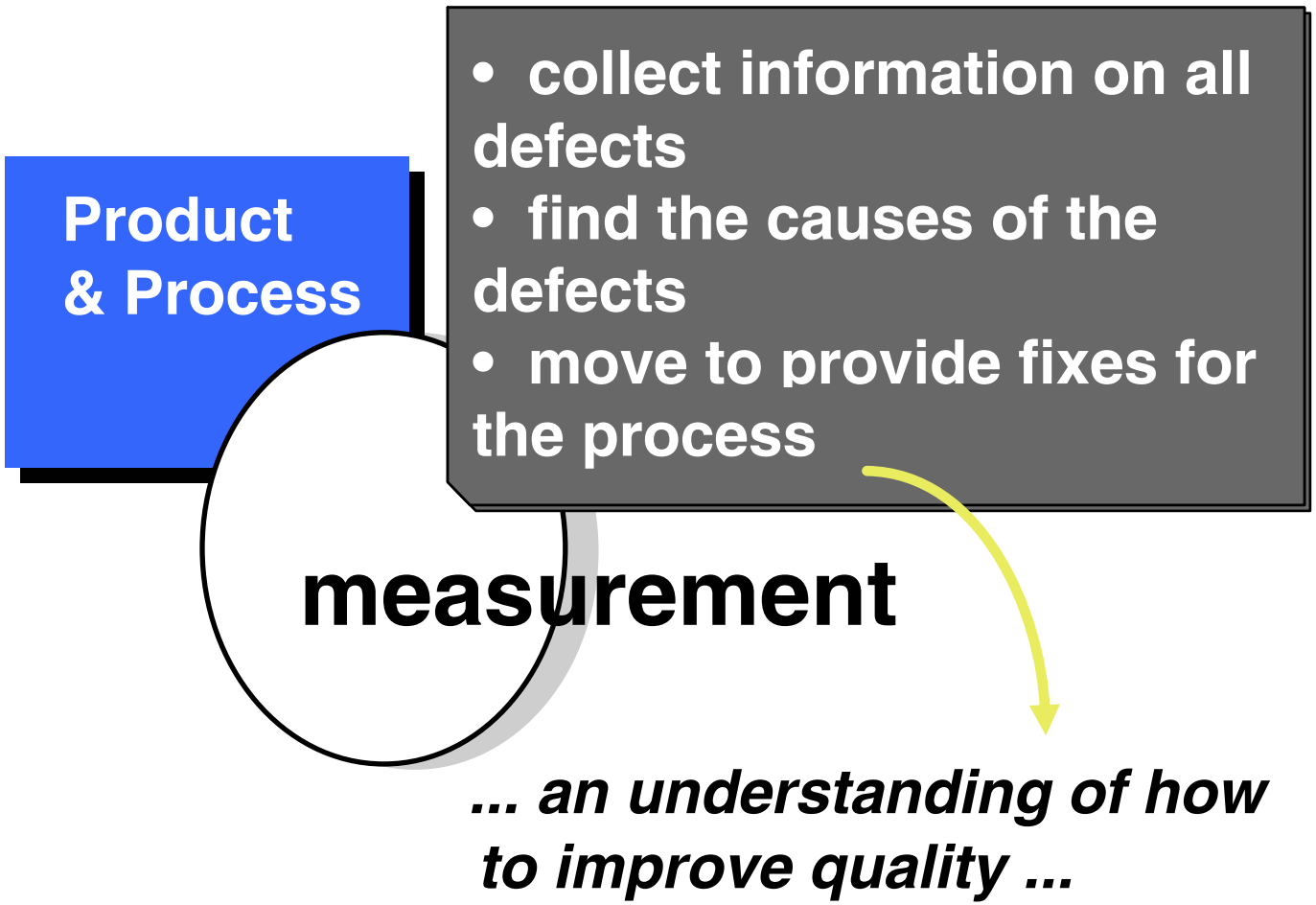
Internal and external attributes



Internal and external attributes

- Some quality attributes (external) are very difficult to measure
- Internal attributes are measured to predict external attributes
- 3 conditions needed:
  - Internal attributes must be measured accurately
  - A relationship must exist between internal ‘measured’ attributes and the external attributes
  - The relation can be expressed in terms of a formula or a model using statistical analysis of available data
  - The relation should be calibrated and validated

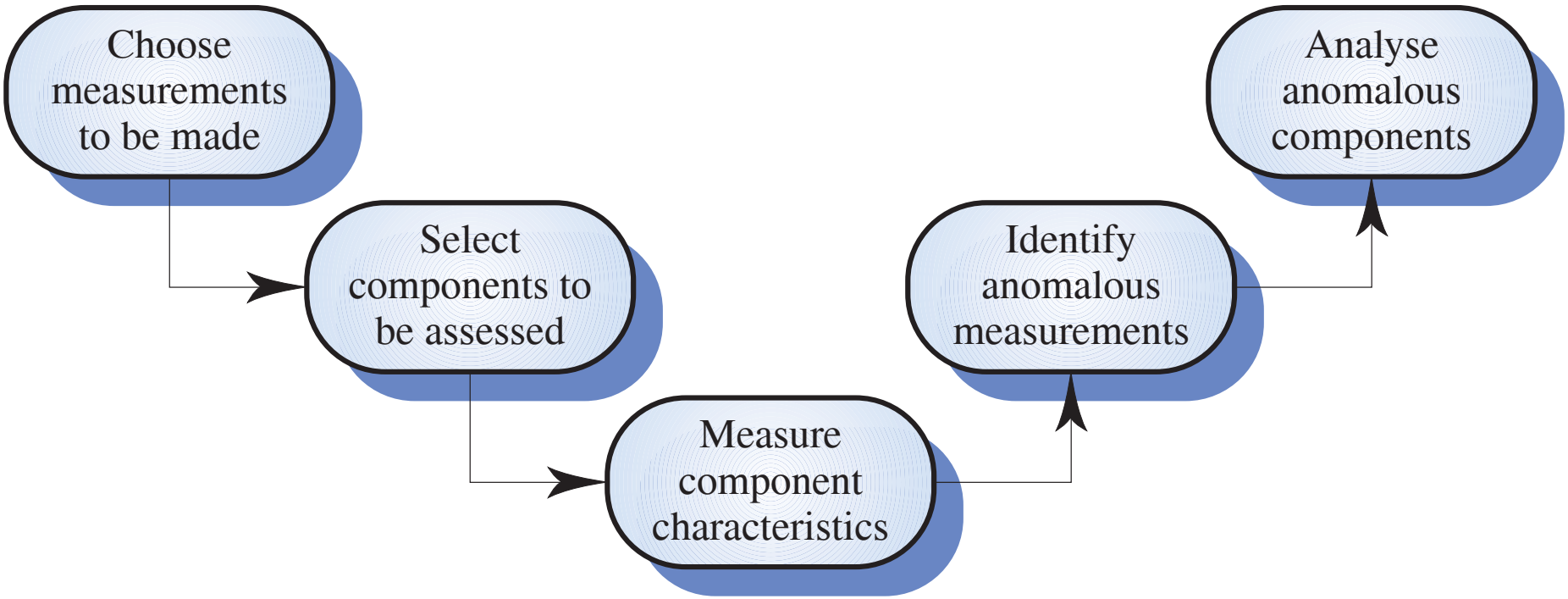
Statistical SQA



The measurement process

- A software measurement process may be part of a quality control process
- Data collected during this process should be maintained as an organisational resource
- Once a measurement database has been established, comparisons across projects become possible

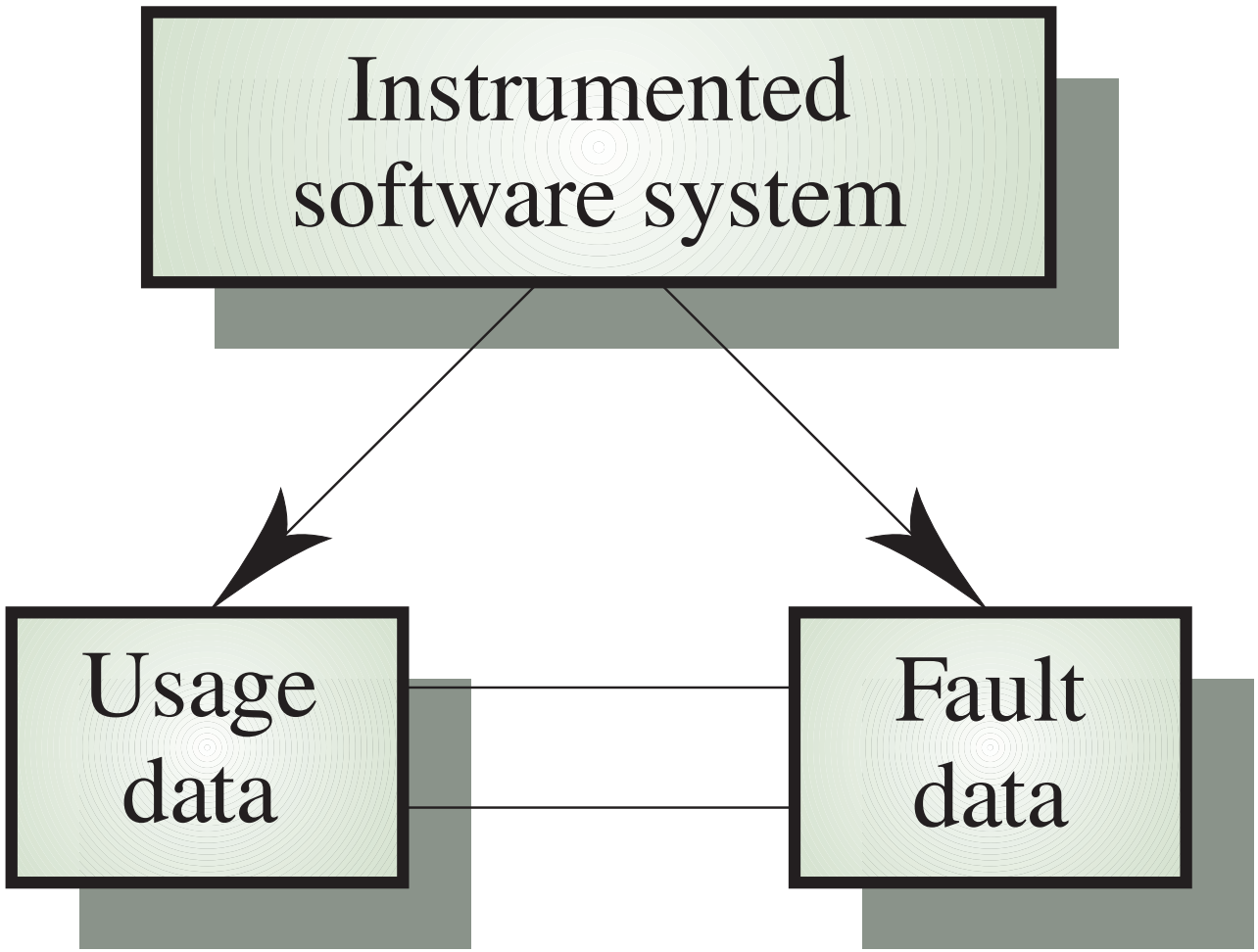
Product measurement process



Data collection

- A metrics programme should be based on a set of product and process data
- Data should be collected immediately (not in retrospect) and, if possible, automatically
- Three types of automatic data collection
  - Static product analysis
  - Dynamic product analysis
  - Process data collation

Automated data collection



Data accuracy

- Don’t collect unnecessary data
  - The questions to be answered should be decided in advance and the required data identified
- Tell people why the data is being collected
  - It should not be part of personnel evaluation
- Don’t rely on memory
  - Collect data when it is generated not after a project has finished

Product metrics

- A quality metric should be a predictor of product quality
- Classes of product metric
  - Dynamic metrics which are collected by measurements made of a program in execution
  - Static metrics which are collected by measurements made of the system representations
  - Dynamic metrics help assess efficiency and reliability; static metrics help assess complexity, understandability and maintainability

Dynamic and static metrics

- Dynamic metrics are closely related to software quality attributes
  - It is relatively easy to measure the response time of a system (performance attribute) or the number of failures (reliability attribute)
- Static metrics have an indirect relationship with quality attributes
  - You need to try and derive a relationship between these metrics and properties such as complexity, understandability and maintainability

Software product metrics

Software metric	Description
Fan in/Fan-out	Fan-in is a measure of the number of functions that call some other function (say X). Fan-out is the number of functions which are called by function X. A high value for fan-in means that X is tightly coupled to the rest of the design and changes to X will have extensive knock-on effects. A high value for fan-out suggests that the overall complexity of X may be high because of the complexity of the control logic needed to coordinate the called components.
Length of code	This is a measure of the size of a program. Generally, the larger the size of the code of a program's components, the more complex and error-prone that component is likely to be.
Cyclomatic complexity	This is a measure of the control complexity of a program. This control complexity may be related to program understandability. The computation of cyclomatic complexity is covered in Chapter 20.
Length of identifiers	This is a measure of the average length of distinct identifiers in a program. The longer the identifiers, the more likely they are to be meaningful and hence the more understandable the program.
Depth of conditional nesting	This is a measure of the depth of nesting of if-statements in a program. Deeply nested if statements are hard to understand and are potentially error-prone.
Fog index	This is a measure of the average length of words and sentences in documents. The higher the value for the Fog index, the more difficult the document may be to understand.

Object-oriented metrics

Object-oriented metric	Description
Depth of inheritance tree	This represents the number of discrete levels in the inheritance tree where sub-classes inherit attributes and operations (methods) from super-classes. The deeper the inheritance tree, the more complex the design as, potentially, many different object classes have to be understood to understand the object classes at the leaves of the tree.
Method fan-in/fan-out	This is directly related to fan-in and fan-out as described above and means essentially the same thing. However, it may be appropriate to make a distinction between calls from other methods within the object and calls from external methods.
Weighted methods per class	This is the number of methods included in a class weighted by the complexity of each method. Therefore, a simple method may have a complexity of 1 and a large and complex method a much higher value. The larger the value for this metric, the more complex the object class. Complex objects are more likely to be more difficult to understand. They may not be logically cohesive so cannot be reused effectively as super-classes in an inheritance tree.
Number of overriding operations	These are the number of operations in a super-class which are over-ridden in a sub-class. A high value for this metric indicates that the super-class used may not be an appropriate parent for the sub-class.

Measurement analysis

- It is not always obvious what data means
  - Analysing collected data is very difficult
- Professional statisticians should be consulted if available
- Data analysis must take local circumstances into account

Measurement surprises

- Reducing the number of faults in a program leads to an increased number of help desk calls
  - The program is now thought of as more reliable and so has a wider more diverse market. The percentage of users who call the help desk may have decreased but the total may increase
  - A more reliable system is used in a different way from a system where users work around the faults. This leads to more help desk calls

Key points

- Software quality management is concerned with ensuring that software meets its required standards
- Quality assurance procedures should be documented in an organisational quality manual
- Software standards are an encapsulation of best practice
- Reviews are the most widely used approach for assessing software quality

Key points

- Software measurement gathers information about both the software process and the software product
- Product quality metrics should be used to identify potentially problematical components
- There are no standardised and universally applicable software metrics