Software Engineering

Lecture-35

Software Testing and Quality Assurance

Topics Covered

- Quality concepts
- Software quality factors
- Software reviews
- The ISO 9001

Reading Assignment

- Roger S. Pressman, "Software Engineering: A Practitioner's Approach," Fifth Edition, McGraw-Hill Book Company Europe, 2001.
  - Chapter 8: Software Quality Assurance

Objectives

- Learn what is software quality assurance (SQA).
- Learn the major quality factors.
- Understand how reviews are conducted.

Software Quality Assurance (SQA)

- SQA encompasses

  (1) a quality management approach,

  (2) effective software engineering technology (methods and tools),

  (3) formal technical reviews that are applied throughout the software process,

  (4) a multi-tiered testing strategy,

  (5) control of software documentation and the changes made to it,

  (6) a procedure to ensure compliance with software development standards (when applicable), and

  (7) measurement and reporting mechanisms.

Software Quality

- **Software quality**:
  - Conformance to explicitly stated requirements and standards
- **Quality assurance**:
  - is the activity that leads to "fitness of purpose".
- **Quality product**:
  - is the one that does what the customer expects it to do.

User satisfaction = compliant product + good quality + delivery within budget and schedule

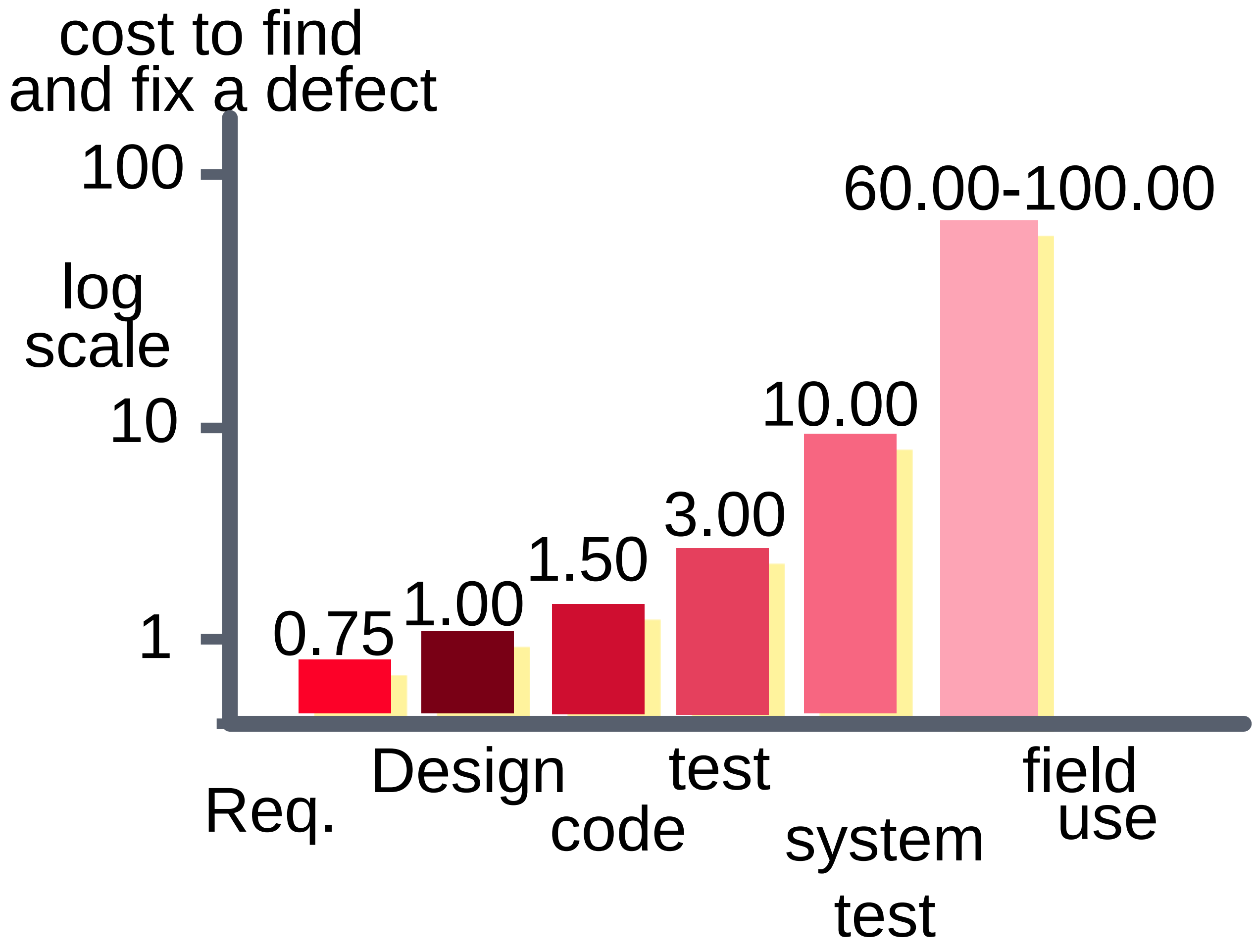
Quality Concepts

- **Quality control**: a series of inspections, reviews, and tests to ensure a product meets the requirements placed upon it.
  - Includes a feedback loop to the process that created the work product.
  - Quality control activities may be fully automated, entirely manual, or a combination of automated tools and human interaction.
- **Quality assurance**: analysis, auditing and reporting activities.
  - provide management with the data necessary to be informed about product quality,

SOFTWARE QUALITY FACTORS

- Functionality, Usability, Reliability, Performance, and Supportability (FURPS) quality factors
  - Functionality: feature set, capabilities, generality of functions, and security
  - Usability: human factors like consistency, and documentation
  - Reliability: frequency and severity of failures, accuracy of outputs, mean time between failures, ability to recover, predictability
  - Performance: processing speed, response time, resource consumption, throughput, and efficiency
  - Supportability: extensibility, adaptability, maintainability, testability, compatibility, configurability

Why SQA Activities Pay Off?



The Quality Movement

- **Total Quality Management** (TQM) is a popular approach for management practice.
- TQM stresses continuous process improvement and can be applied to software development.
- Not much can be done to improve quality until a visible, repeatable, and measurable process is created.

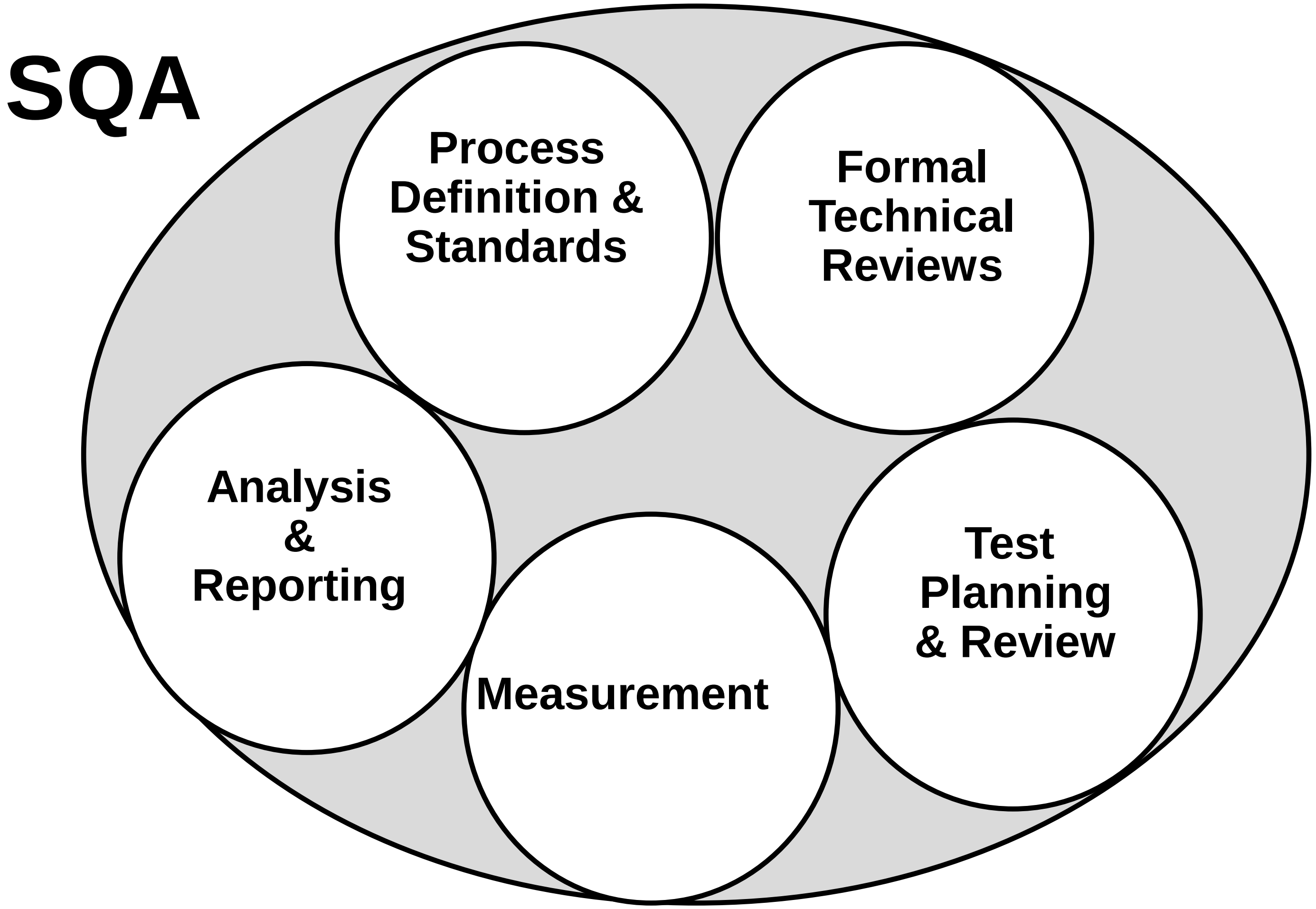
TQM

1. Refers to a system of continuous process improvement.
   - develop a process that is visible, repeatable, and measurable.
2. *This* step examines intangibles that affect the process and works to optimize their impact on the process.
3. This step concentrates on the user of the product. Examining the way the user applies the product. This step leads to improvement in the product itself and, potentially, to the process that created it.
4. This is a business-oriented step that looks for opportunity in related areas identified by observing the use of the product in the marketplace.

Software Quality Assurance

- The SQA group must look at software from the customer's perspective, as well as assessing its technical merits.
- The activities performed by the SQA group involve quality planning, oversight, record keeping, analysis and reporting.

Software Quality Assurance (Cont.)



Software Quality Assurance (Cont.)

- Beginning of the project
  - Project manager will consider quality factors and decide which ones are important for the system
  - Decide on what validation and verification activities will be carried out to check that the required quality factors are present in the product
- During the project
  - Validation and verification of quality standards and procedures
- End of the project
  - Expected quality achieved to what extent

Software Reviews

- Any work product (including documents) should be reviewed.
- Conducting timely reviews of all work products can often eliminate 80% of the defects before any testing is conducted.
- This message often needs to be carried to managers in the field, whose impatience to generate code sometimes makes them reluctant to spend time on reviews.

WHAT ARE REVIEWS AND WHAT REVIEWS ARE NOT?

- Reviews are:
  - 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
- Reviews are not:
  - A project budget summary
  - A scheduling assessment
  - An overall progress report

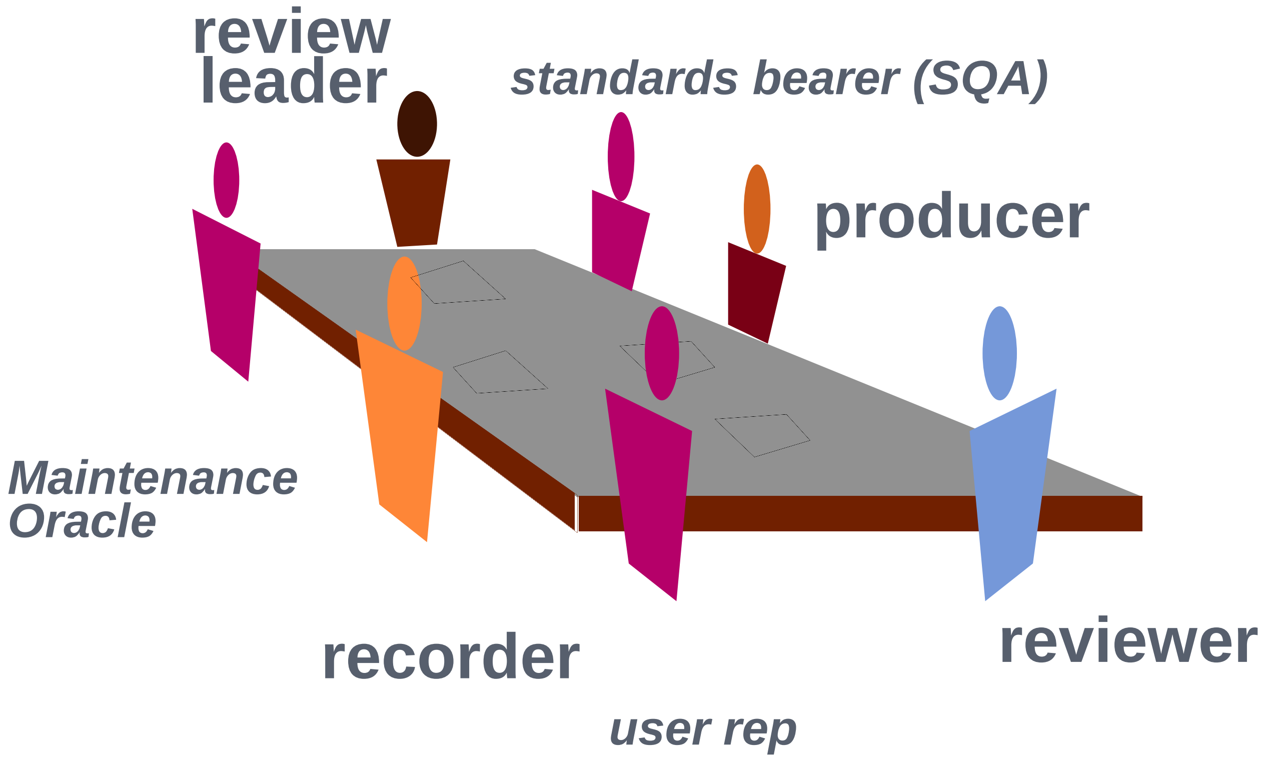
Formal Technical Reviews (FTR)

- The objectives of FTR are:
  - To uncover errors in functions, logic, or implementation for any representation of the software.
  - To verify that the software under review meets its requirements.
  - To ensure that the software has been represented according to predefined standards.
  - To achieve software that is developed in a uniform manner.
  - To make projects more manageable.

Notes on Formal Technical Reviews

- Review *the product,* not the producer
- Set an agenda and maintain it
- Take written notes
- Limit the number of participants and insist upon advance preparation
- The duration of the review meeting should be less than two hours.
- Develop a checklist for each product that is likely to be reviewed

The Players


review leader
standards bearer (SQA)
producer
Maintenance Oracle
recorder
reviewer
user rep

How an FTR is Performed

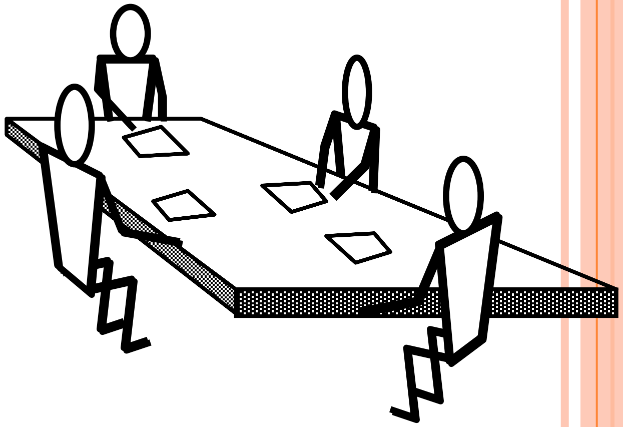
- Producer informs the project leader that the product is complete and that a review is required
- The project leader forms a review team and appoints a review leader
- The review leader evaluate the product for readiness, generates copies of the product material, distributes to the reviewers, and schedules a review meeting
- Each reviewer reviews the product. He becomes familiar with the product and makes notes of concerns

How an FTR is Performed (Cont.)

- Review leader, all reviewers, and producer attend the meeting
- One of the reviewers take the role of recorder
- During the meeting, the producer walks through the product, explaining the material, while the reviewers raise issue based on their preparation. If an error is discovered, then it is recorded by the recorder.

Conducting the Review

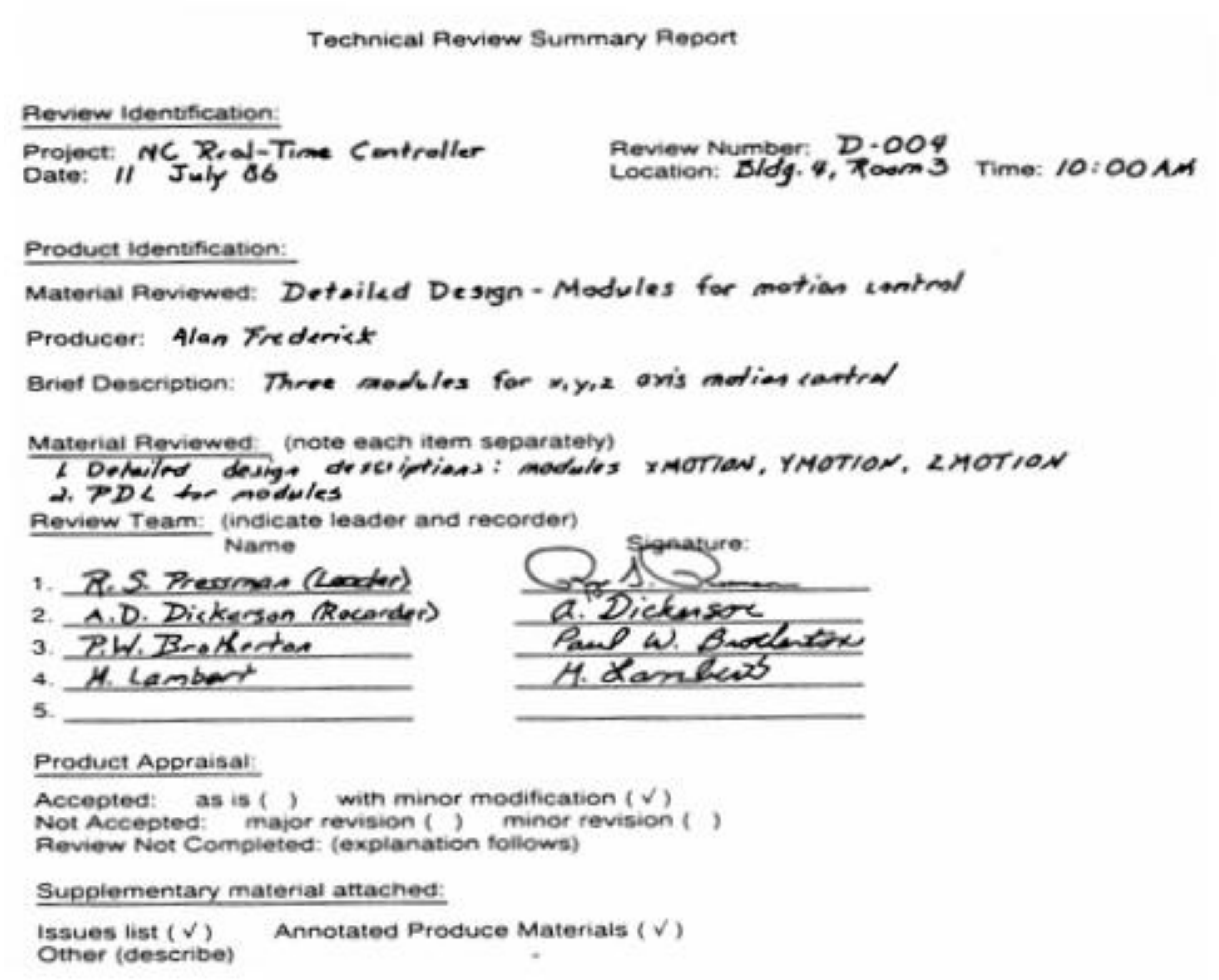
1. Be prepared—evaluate product before the review
2. Review the product, not the producer
3. Keep your tone mild, ask questions instead of making accusations
4. Stick to the review agenda
5. Raise issues, don't resolve them
6. Avoid discussions of style—stick to technical correctness
7. Schedule reviews as project tasks
8. Record and report all review results



Outcome of an FTR

- The attendees of the meeting decide whether to:
  - Accept the product without further modification
  - Accept the product provisionally. Minor errors have been encountered. These must be fixed but no additional review will be needed
  - Reject the product due to sever errors. Once the errors are fixed, another review should be conducted
- At the end of an FTR, a review summary report should be produced. It should answer the following
  - What was reviewed?
  - Who was involved in the review?
  - What were the findings and conclusions?

Outcome of an FTR (Cont.)



Technical Review Summary Report

Review Identification:

Project: NC Real-Time Controller
Date: 11 July 86

Review Number: D-004
Location: Bldg. 4, Room 3   Time: 10:00 AM

Product Identification:

Material Reviewed: Detailed Design - Modules for motion control

Producer: Alan Frederick

Brief Description: Three modules for x,y,z axis motion control

Material Reviewed: (note each item separately)
1. Detailed design descriptions: modules XMOTION, YMOTION, ZMOTION
2. PDL for modules

Review Team: (indicate leader and recorder)

	Name	Signature:
1.	R.S. Pressman (Leader)	[illegible]
2.	A.D. Dickerson (Recorder)	A. Dickerson
3.	P.W. Brotherton	Paul W. Brotherton
4.	H. Lambert	H. Lambert
5.		

Product Appraisal:

Accepted: as is ( ) with minor modification (✓)
Not Accepted: major revision ( ) minor revision ( )
Review Not Completed: (explanation follows)

Supplementary material attached:

Issues list (✓) Annotated Produce Materials (✓)
Other (describe)

REVIEW CHECKLISTS

- FTR can be conducted during each step in the software engineering process.
- Checklists can be used to assess products that are derived as part of software development.
- The checklists are not intended to be comprehensive, but rather to provide a point of departure for each review.

Software Project Planning

- Software project planning develops estimates for resources, cost and schedule based on the software allocation established as part of the system engineering activity.
- Like any estimation process, software project planning is inherently risky.
- The review of the Software Project Plan establishes the degree of risk.

SOFTWARE PROJECT PLANNING (CONT.)

- The following checklist is applicable:
  - Is software scope unambiguously defined and bounded?
  - Is terminology clear?
  - Are resources adequate for scope?
  - Are resources readily available?
  - Have risks in all important categories been defined.
  - Is a risk management plan in place?
  - Are tasks properly defined and sequenced?

Software Project Planning (Cont.)

- Is parallelism reasonable given available resources?
- Is the basis for cost estimation reasonable? Has the cost estimate been developed using two independent methods?
- Have historical productivity and quality data been used?
- Have differences in estimates been reconciled?
- Are pre-established budgets and deadlines realistic?
- Is the schedule consistent?

Software Requirements Analysis

- Reviews for software requirements analysis focus on traceability to system requirements and consistency and correctness of the analysis model.
- A number of FTRs are conducted for the requirements of a large system and may be also followed by reviews and evaluation of prototypes as well as customer meetings.

Software Requirements Analysis (Cont.)

- The following topics are considered during FTRs for analysis:
  - Is information domain analysis complete, consistent and accurate?
  - Is problem partitioning complete?
  - Are external and internal interfaces properly defined?
  - Does the data model properly reflect data objects, their attributes and relationships.
  - Are all requirements traceable to system level?

Software Requirements Analysis (Cont.)

- Has prototyping been conducted for the user/customer?
- Is performance achievable within the constraints imposed by other system elements?
- Are requirements consistent with schedule, resources and budget?
- Are validation criteria complete?

SOFTWARE DESIGN

- Reviews for software design focus on data design, architectural design and procedural design.
- In general, two types of design reviews are conducted:
  - The preliminary design review assesses the translation of requirements to the design of data and architecture.
  - The second review, often called a design walkthrough, concentrates on the procedural correctness of algorithms as they are implemented within program modules.

Software Design (Cont.)

- The following checklists are useful for preliminary design review:
  - Are software requirements reflected in the software architecture?
  - Is effective modularity achieved? Are modules functionally independent?
  - Are interfaces defined for modules and external system elements?
  - Is the data structure consistent with information domain?
  - Is data structure consistent with software requirements?
  - Has maintainability been considered?
  - Have quality factors been explicitly assessed?

Software Design (Cont.)

- The following checklists are useful for Design walkthrough:
  - Does the algorithm accomplishes desired function?
  - Is the algorithm logically correct?
  - Is the interface consistent with architectural design?
  - Is the logical complexity reasonable?
  - Has error handling been specified?
  - Are local data structures properly defined?

Software Design (Cont.)

- Are structured programming constructs used throughout?
- Is design detail amenable to implementation language?
- Which are used: operating system or language dependent features?
- Has maintainability been considered?

CODING

- Errors can be introduced as the design is translated into a programming language.
- A code walkthrough can be an effective means for uncovering these translation errors.

Coding (Cont.)

- The following checklist assumes that a design walkthrough has been conducted and that algorithm correctness has been established as part of the design FTR.
  - Has the design properly been translated into code? [The results of the procedural design should be available during this review.]
  - Are there misspellings and typos?
  - Has proper use of language conventions been made?
  - Is there compliance with coding standards for language style, comments, module prologue?

Coding (Cont.)

- Are there incorrect or ambiguous comments?
- Are data types and data declaration proper?
- Are physical constants correct?
- Have all items on the design walkthrough checklist been re-applied (as required)?

Software Testing

- Software testing is a quality assurance activity in its own right.
- The completeness and effectiveness of testing can be dramatically improved by critically assessing any test plans and procedures that have been created.

TEST PLAN

- The following checklists are useful for test plan walkthrough:
  - Have major test phases properly been identified and sequenced?
  - Has traceability to validation criteria/requirements been established as part of software requirements analysis?
  - Are major functions demonstrated early?
  - Is the test plan consistent with overall project plan?
  - Has a test schedule been explicitly defined?
  - Are test resources and tools identified and available?
  - Has a test record keeping mechanism been established?

Test Procedure

- The following checklists are useful for test procedure walkthrough:
  - Have both white and black box tests been specified?
  - Have all independent logic paths been tested?
  - Have test cases been identified and listed with expected results?
  - Is error-handling to be tested?
  - Have all boundary values been tested?
  - Are timing and performance to be tested?
  - Has acceptable variation from expected results been specified?

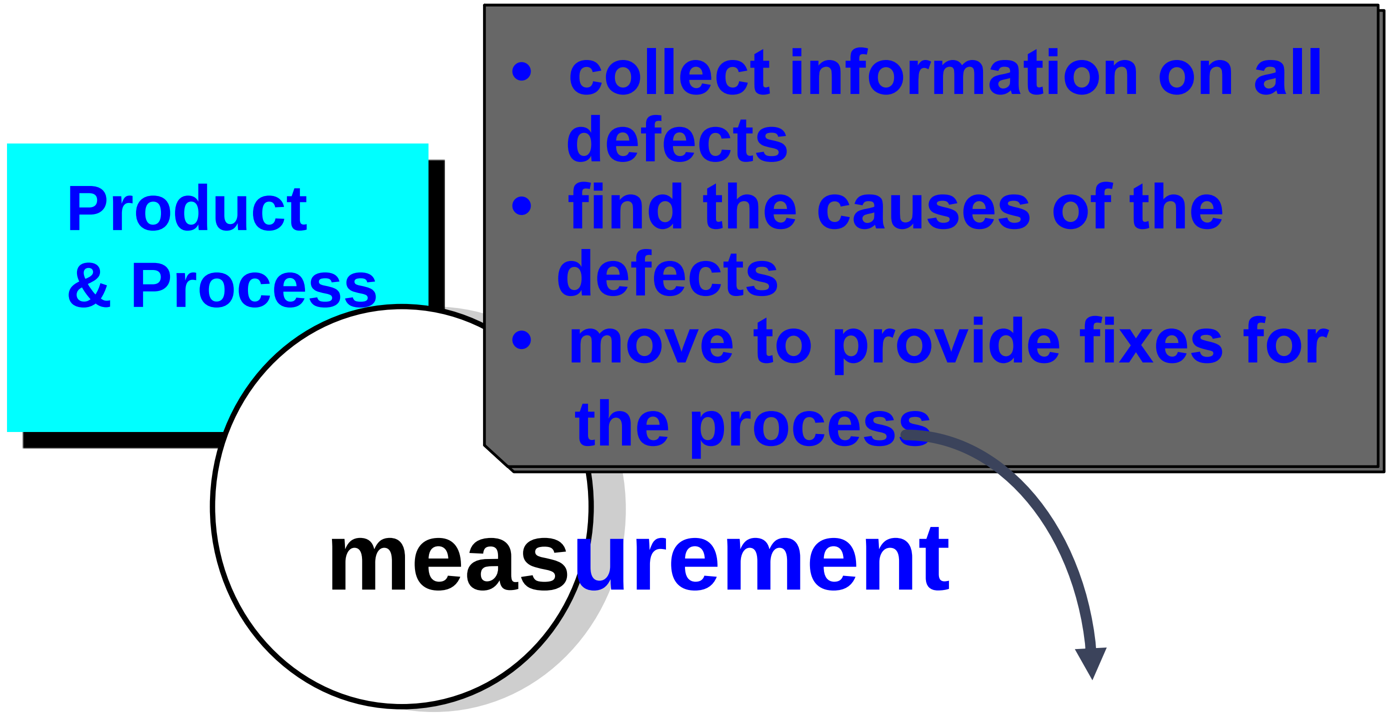
Maintenance

- The review checklists for software development are equally valid for the software maintenance phase. In addition to all of the questions posed in the checklists, the following special considerations should be kept in mind:
  - Have side effects associated with change been considered?
  - Has the request for change been documented, evaluated and approved?
  - Has the change, once made, been documented and reported to interested parties?
  - Have appropriate FTRs been conducted?
  - Has a final acceptance review been conducted to ensure that all software has been properly updated, tested and replaced?

Statistical Quality Assurance

- Each defect needs to be traced to its cause.
- Defect causes having the greatest impact on the success of the project must be addressed first.

Statistical SQA



- collect information on all defects
- find the causes of the defects
- move to provide fixes for the process

... an understanding of how to improve quality ...

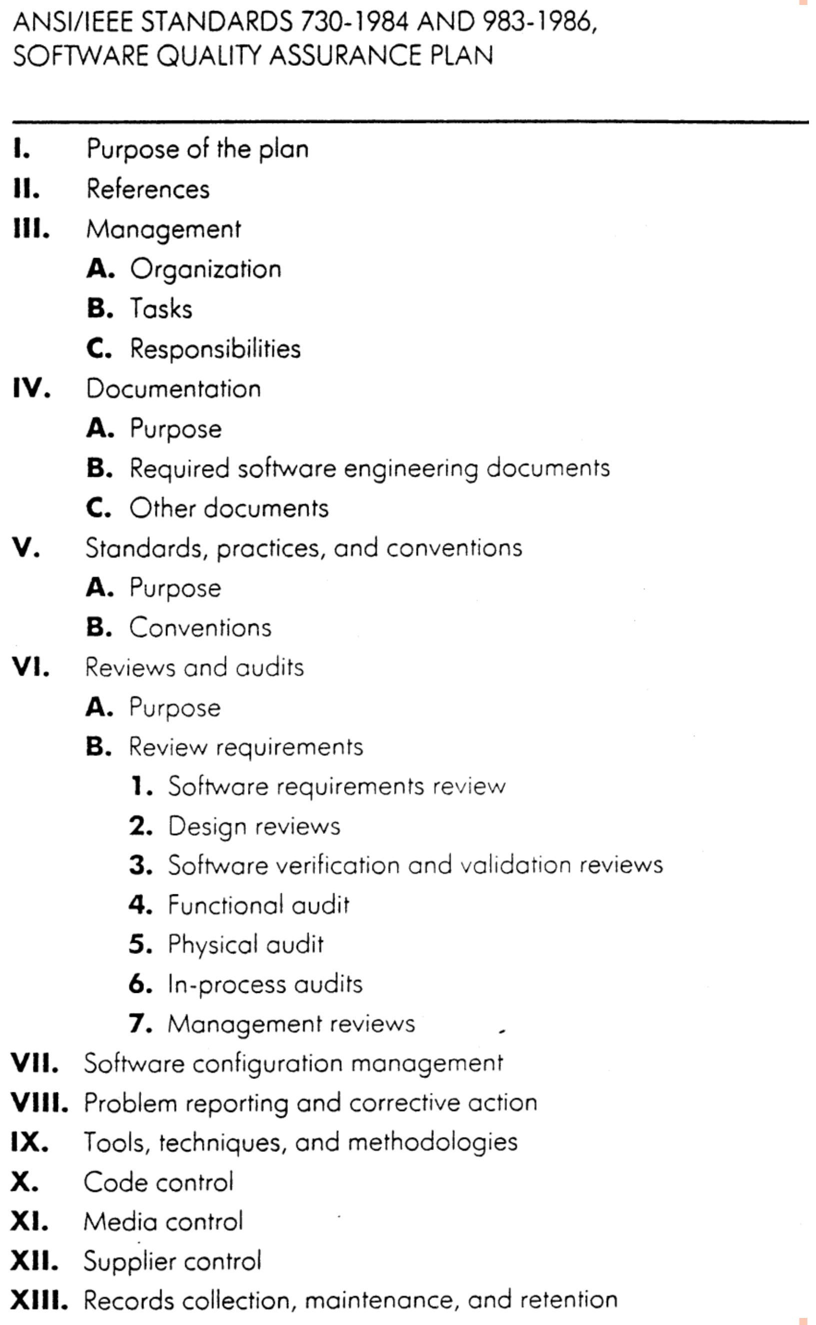
Metrics Derived from Reviews

- 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 requirements)
- Number of errors found during preparation

Software Reliability

- **Software consistency**: repeatability of results.
- **Reliability**: probability of failure free operation for a specified time period.
  - Don’t apply hardware reliability theory to software (e.g. a key point is that, unlike hardware, software does not wear out so that failures are likely to be caused by design defects).

IEEE STANDARD: SOFT QUALITY ASSURANCE P



ANSI/IEEE STANDARDS 730-1984 AND 983-1986,
SOFTWARE QUALITY ASSURANCE PLAN

- **I.** Purpose of the plan
- **II.** References
- **III.** Management
  - **A.** Organization
  - **B.** Tasks
  - **C.** Responsibilities
- **IV.** Documentation
  - **A.** Purpose
  - **B.** Required software engineering documents
  - **C.** Other documents
- **V.** Standards, practices, and conventions
  - **A.** Purpose
  - **B.** Conventions
- **VI.** Reviews and audits
  - **A.** Purpose
  - **B.** Review requirements
    - **1.** Software requirements review
    - **2.** Design reviews
    - **3.** Software verification and validation reviews
    - **4.** Functional audit
    - **5.** Physical audit
    - **6.** In-process audits
    - **7.** Management reviews
- **VII.** Software configuration management
- **VIII.** Problem reporting and corrective action
- **IX.** Tools, techniques, and methodologies
- **X.** Code control
- **XI.** Media control
- **XII.** Supplier control
- **XIII.** Records collection, maintenance, and retention

The ISO 9001 Quality Standards

- One of the most important worldwide standards for quality assurance
- Adopted for use by over 130 countries
- Not industry specific but expressed in general terms
- ISO 9001 is the quality assurance standard that contains 20 requirements that must be present in any software quality assurance system.

The ISO 9001: How it Works?

- A software developer implements a quality system according to ISO 9001 specifications
- The quality system is used for some time to detect any problems in the system
- Third party audit
- Accreditation request to the responsible body
- Accreditation body inspect the quality system documentation and visits the organization
- On successful inspection, a certificate is issued
- Unannounced visits to check whether the quality system is adhered to or not

KEY POINTS

- Quality assurance: is the activity that leads to “fitness of purpose”
- Major software quality factors: functionality, usability, reliability, performance, and supportability.
- Any work product (including documents) should be reviewed.
- FTR are to find errors during the process so that they don’t become defects after the release of the software.
- Checklists can be used to assess products that are derived as part of software development.
- ISO 9001 is the quality assurance standard that contains 20 requirements that must be present in any software quality assurance system.