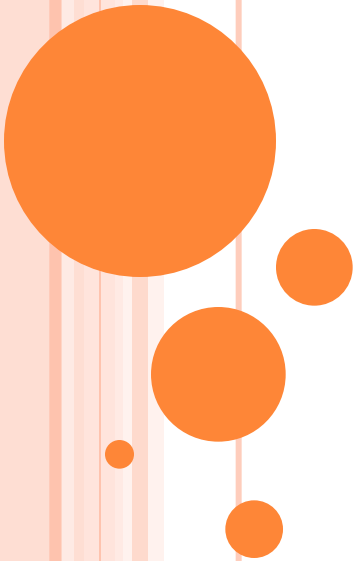
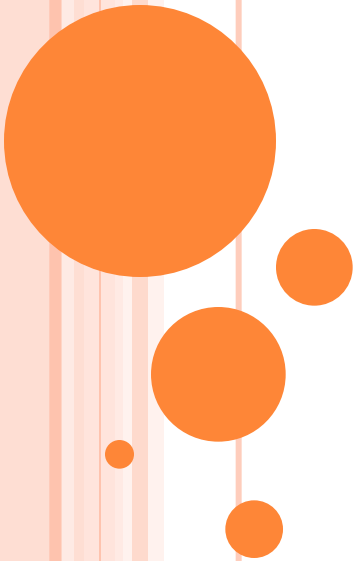
Software Engineering



LECTURE-38



Software Quality Assurance

Topics Covered

- Quality Concepts
- Quality Cost
- Total Quality Management
- Software Reviews

Quality Concepts - 1

- Variation control is the heart of quality control
- Software engineers strive to control the
  - process applied
  - resources expended
  - end product quality attributes
- Quality of design
  - refers to characteristics designers specify for the end product to be constructed

Quality Concepts - 2

- Quality of conformance
  - degree to which design specifications are followed in manufacturing the product
- Quality control
  - series of inspections, reviews, and tests used to ensure conformance of a work product to its specifications
- Quality assurance
  - auditing and reporting procedures used to provide management with data needed to make proactive decisions

Quality Costs

- quality planning, formal technical reviews, test equipment, training

- Appraisal costs
  - in-process and inter-process inspection, equipment calibration and maintenance, testing
- Failure costs
  - rework, repair, failure mode analysis
- External failure costs
  - complaint resolution, product return and replacement, help line support, warranty work

Total Quality Management - 1

- Kaizen
  - develop a process that is visible, repeatable, and measurable
- Atarimae hinshitsu
  - examine the intangibles that affect the process and work to optimize their impact on the process

Total Quality Management - 2

- Kanse
  - examine the way the product is used by the customer with an eye to improving both the product and the development process
- Miryokuteki hinshitsu
  - observe product use in the market place to uncover new product applications and identify new products to develop

Software Quality Assurance

- Conformance to software requirements is the foundation from which software quality is measured.
- Specified standards are used to define the development criteria that are used to guide the manner in which software is engineered.
- Software must conform to implicit requirements (ease of use, maintainability, reliability, etc.) as well as its explicit requirements.

SQA Group Activities - 1

- Prepare SQA plan for the project.
- Participate in the development of the project's software process description.
- Review software engineering activities to verify compliance with the defined software process.

SQA Group Activities - 2

- Audit designated software work products to verify compliance with those defined as part of the software process.
- Ensure that any deviations in software or work products are documented and handled according to a documented procedure.
- Record any evidence of noncompliance and reports them to management.

Software Reviews

- Purpose is to find defects (errors) before they are passed on to another software engineering activity or released to the customer.
- Software engineers (and others) conduct formal technical reviews (FTR) for software engineers.
- Using formal technical reviews (walkthroughs or inspections) is an effective means for improving software quality.

Review Roles

- maintenance oracle
- standards bearer
- user representative
- others

Formal Technical Reviews - 1

- Involves 3 to 5 people (including reviewers)
- Advance preparation (no more than 2 hours per person) required
- Duration of review meeting should be less than 2 hours
- Focus of review is on a discrete work product
- Review leader organizes the review meeting at the producer's request.

Formal Technical Reviews - 2

- Reviewers ask questions that enable the producer to discover his or her own error (the product is under review not the producer)
- Producer of the work product walks the reviewers through the product
- Recorder writes down any significant issues raised during the review
- Reviewers decide to accept or reject the work product and whether to require additional reviews of product or not.

Why do peer reviews?

- To improve quality.
- Catches 80% of all errors if done properly.
- Catches both coding errors and design errors.
- Enforce the spirit of any organization standards.
- Training and insurance.

Formality and Timing

- Formal review presentations
  - resemble conference presentations.
- Informal presentations
  - less detailed, but equally correct.
- Early
  - tend to be informal
  - may not have enough information
- Later
  - tend to be more formal
  - Feedback may come too late to avoid rework

Formality and Timing

- Analysis is complete.
- Design is complete.
- After first compilation.
- After first test run.
- After all test runs.
- Any time you complete an activity that produce a complete work product.

Review Guidelines

- Keep it short (< 30 minutes).
- Don’t schedule two in a row.
- Don’t review product fragments.
- Use standards to avoid style disagreements.
- Let the coordinator run the meeting and maintain order.

Formal SQA Approaches

1. Proof of correctness.
2. Statistical quality assurance.
3. Cleanroom process combines items 1 & 2.

Statistical Quality Assurance

- Information about software defects is collected and categorized
- Each defect is traced back to its cause
- Using the Pareto principle (80% of the defects can be traced to 20% of the causes) isolate the "vital few" defect causes
- Move to correct the problems that caused the defects

Software Reliability

- Defined as the probability of failure free operation of a computer program in a specified environment for a specified time period
- Can be measured directly and estimated using historical and developmental data (unlike many other software quality factors)
- Software reliability problems can usually be traced back to errors in design or implementation.

Software Reliability Metrics

- Reliability metrics are units of measure for system reliability
- System reliability is measured by counting the number of operational failures and relating these to demands made on the system at the time of failure
- A long-term measurement program is required to assess the reliability of critical systems

Reliability Metrics - part 1

- Probability of Failure on Demand (POFOD)
  - POFOD = 0.001
  - For one in every 1000 requests the service fails per time unit
- Rate of Fault Occurrence (ROCOF)
  - ROCOF = 0.02
  - Two failures for each 100 operational time units of operation

Reliability Metrics - part 2

- Mean Time to Failure (MTTF)
  - average time between observed failures (aka MTBF)
- Availability = MTBF / (MTBF+MTTR)
  - MTBF = Mean Time Between Failure
  - MTTR = Mean Time to Repair
- Reliability = MTBF / (1+MTBF)

Time Units

- Raw Execution Time
  - non-stop system
- Calendar Time
  - If the system has regular usage patterns
- Number of Transactions
  - demand type transaction systems

Software Safety

- SQA activity that focuses on identifying potential hazards that may cause a software system to fail.
- Early identification of software hazards allows developers to specify design features to can eliminate or at least control the impact of potential hazards.
- Software reliability involves determining the likelihood that a failure will occur without regard to consequences of failures.

Validation Perspectives

- Reliability validation
  - Does measured system reliability meet its specification?
  - Is system reliability good enough to satisfy users?
- Safety validation
  - Does system operate so that accidents do not occur?
  - Are accident consequences minimized?
- Security validation
  - Is system secure against external attack?

Validation Techniques

- Static techniques
  - design reviews and program inspections
  - mathematical arguments and proof
- Dynamic techniques
  - statistical testing
  - scenario-based testing
  - run-time checking
- Process validation
  - SE processes should minimize the chances of introducing system defects

Static Validation Techniques

- Concerned with analysis of documentation
- Focus is on finding system errors and identifying potential problems that may arise during system operation
- Documents may be prepared to support static validation
  - structured arguments
  - mathematical proofs

Static Safety Validation Techniques

- Demonstrating safety by testing is difficult
- Testing all possible operational situations is impossible
- Normal reviews for correctness may be supplemented by specific techniques intended to make sure unsafe situations never arise

Safety Reviews

- Intended system functions correct?
- Is structure maintainable and understandable?
- Verify algorithm and data structure design against specification
- Check code consistency with algorithm and data structure design
- Review adequacy of system testing

Hazard-Driven Analysis

- Effective safety assurance relies on hazard identification
- Safety can be assured by
  - hazard avoidance
  - accident avoidance
  - protection systems
- Safety reviews should demonstrate that one or more of these techniques have been applied to all identified hazards

System Safety Case

- The normal practice for a formal safety case to be required for all safety-critical computer-based systems
- A safety case presents a list of arguments, based on identified hazards, as to why there is an acceptably low probability that these hazards will not result in an accident
- Arguments can be based on formal proof, design rationale, safety proofs, and process factors

Poka-Yoke Devices

- Mechanisms that lead to the prevention of a potential quality problem before it occurs or to the rapid detection of a quality problem if one is introduced
- Are a simple, cheap, part of the engineering process, and are located near the process task where the mistakes are likely to occur

SQA Plan – 1

- Management section
  - describes the place of SQA in the structure of the organization
- Documentation section
  - describes each work product produced as part of the software process
- Standards, practices, and conventions section
  - lists all applicable standards/practices applied during the software process and any metrics to be collected as part of the software engineering work

SQA Plan - 2

- Reviews and audits section
  - provides an overview of the approach used in the reviews and audits to be conducted during the project
- Test section
  - references the test plan and procedure document and defines test record keeping requirements

SQA Plan - 3

- Problem reporting and corrective action section
  - defines procedures for reporting, tracking, and resolving errors or defects, identifies organizational responsibilities for these activities
- Other
  - tools, SQA methods, change control, record keeping, training, and risk management