SOFTWARE QUALITY ASSURANCE

Lecture 3

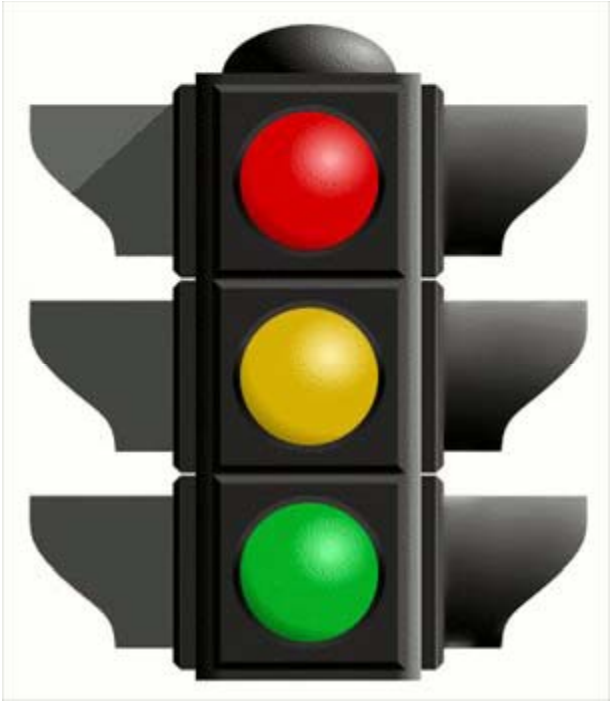
Instructor: Mr. Natash Ali Mian

Department of CS and IT
The University of Lahore

Switch off mobile phones during lectures, or put them into silent mode

CONTENTS

- Term Paper
- Characteristics of good and bad quality
- Project Management and Software Quality
- SQA Plan
- Economics of Software Quality
- Quality Measurements



Please Obey Traffic Signals

Donate **blood**

to save lives

KEEP YOUR SURROUNDING CLEAN

Term Paper

Finalize Group Members	26-Feb-2013
Finalize Topic	**12-Mar-2013**
Search Papers and Sort Selected (At least 15)	19-Mar-2013
Go Through the Abstract and Introduction of Selected Papers	26-Mar-2013
Submit a Summary and Comments on related papers	**TBD**
Submit Initial Draft	**TBD**
Final Paper Submission	**TBD**
Final Presentation	**TBD**

Please note that Every Phase has Marks

How to find Topics???
What are the latest research trends?

Challenges

- Challenges
  - Dealing with increasing complexity of software processes and products
  - Defining improved quality models
  - Integrating better with development processes
  - Increasing the evidence that standards that promote higher quality software bring indeed significant benefits
  - Finding out ways to compose standards
  - Improving QA models and processes
  - Increasing software quality-centric adaptability
  - Increasing SQA independence from software development
  - Dealing with the perception that QA is less glamorous than development

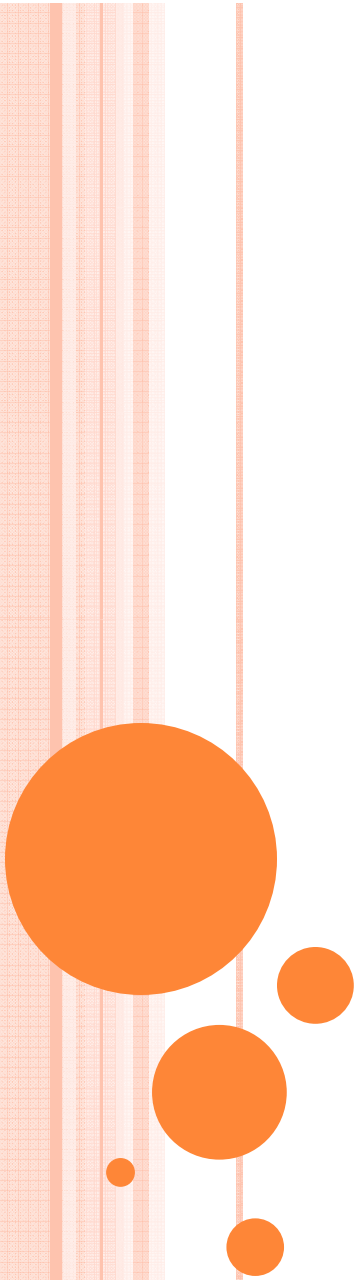
Trends in software quality

- Metrics and measurements
- Methods and tools
- Testing effectiveness and code coverage
- Inspections, reviews, and walkthroughs
- Software architectures
- Management and certification

Research directions

- QA for COTS-based software
- QA for agile development processes
- QA for open-source software
- Metrics for software reliability prediction
- 100% automatic testing
- Personal and Team software processes

Lets now move to the theory part



Defect Prevention

Defect Prevention

- We do not want defects or faults to enter our work products, requirements, design, code, or other documents
- We try to eliminate the error sources in defect prevention
- Defect prevention is very difficult to understand, study, and quantify

Philosophy of Defect Prevention - 1

- If human misconceptions are the error sources, education and training can help us remove these error sources

- If imprecise designs and implementations that deviate from product specifications or design intentions are the causes for faults, formal methods can help us prevent such deviations

Philosophy of Defect Prevention - 2

- If non-conformance to selected processes or standards is the problem that leads to fault injections, then process conformance or standard enforcement can help use prevent the injection of related faults

- If certain tools or technologies can reduce fault injections under similar environments, they should be adopted

Education and Training - 1

- Education and training provide people-based solutions for error source elimination
- The people factor is the most important factor that determines the quality and, ultimately, the success or failure of most software projects

EDUCATION AND TRAINING - 2

- Education and training of software professionals can help them control, manage, and improve the way they work

Focus of Education & Training

- Product and domain specific knowledge
- Software development knowledge and expertise
- Knowledge about Development methodology, technology, and tools
- Development process knowledge

Product and Domain Specific Knowledge

- If the people involved are not familiar with the product type or application domain, there is a good chance that wrong solutions will be implemented

Formal Methods

- Formal methods provide a way to eliminate certain error sources and to verify the absence of related faults
- Formal methods include
  - Formal specification
  - Formal verification

Formal Specification

- Formal specification is concerned with producing an unambiguous set of product specifications so that
  - customer requirements,
  - environmental constraints
  - design intentions,

- are correctly reflected, thus reducing the chances of accidental fault injections

Formal Verifications

- Formal verification checks
  - the conformance of software design or code against these formal specifications,
- thus ensuring that the software is fault free with respect to its formal specifications

Root Causes of Poor Software Quality

Root Causes of Poor Software Quality - 1

- Inadequate training of managers and staff
- Inadequate defect and cost measurement
- Excessive schedule pressure

Root Causes of Poor Software Quality - 2

- Insufficient defect removal
- High complexity levels
- Ambiguous and creeping requirements and design (*feature race & gimmicks*)

INDUSTRY STATISTICS

- Lets look at the status of the software industry's seriousness of the software industry with respect to the software quality assurance

Quality Assurance Organizations

- No quality assurance 60%
- Token quality assurance 20%
- Passive quality assurance 15%
- Active quality assurance 5%

POINT TO REMEMBER

- There is another point that must be remembered that software varies from industry to industry
- The focus on software quality naturally is dependent on the industry, as well as the importance of the software application.
- More critical applications, naturally, need to have higher software quality than others

Software Quality in Six Sub- Industries

- Systems software that controls physical devices
- Information systems that companies build for their own use
- Outsource or contract software built for clients
- Commercial software built by vendors for lease or sale
- Military software built following various military standards
- End-user software built for private use by computer literate workers or managers

Characteristics of Quality Laggards

- We'll now discuss the characteristics of companies, which produce poor quality software

QUALITY LAGGARDS - 1

- No software quality measurement program of any kind
- No usage of formal design and code inspections
- No knowledge of the concepts of defect potentials and defect removal efficiency

Quality Laggards - 2

- Either no quality assurance group or a group that is severely understaffed
- No trained testing specialists available
- Few or no automated quality assurance tools
- No quality and reliability estimation capability

Quality Laggards - 3

- Minimal or no software project management tools available
- No automated risk assessment or avoidance capability
- From a low of one to a high of perhaps four distinct testing stages

Quality Laggards - 4

- No test library or test-case management tools available
- No complexity analysis tools utilized
- Defect potentials averaging more than 6 defects per function point

Quality Laggards - 5

- Defect removal efficiency averaging less than 80%
- Executive and managerial indifference (and ignorance) of quality matters

Other Related Issues

- Staff morale and voluntary attrition
- Market shares and competitive positioning
- Litigation and product recalls

QUALITY LAGGARDS

- Quality laggards are the companies with highest probability of cancelled projects, several schedule overruns, and severe overruns
- It is no coincidence that software groups among the quality laggards also tend to be candidates for immediate replacement by outsource organizations

Categories of Quality Attributes

- Product-specific quality attributes
- Organization-specific quality attributes

Product-Specific Attributes - 1

- Ease of use
- Documentation
- Defect tolerance
- Defect frequency

Product-Specific Attributes - 2

- Defect impact
- Packaging
- Price versus reliability
- Performance

Organization-Specific Attributes

- Service and support
- Internal processes

Achieving High Levels of Software Quality - 1

- Enterprise-wide quality programs
- Quality awareness and training methods
- Quality standards and guidelines
- Quality analysis methods
- Quality measurement methods

Achieving High Levels of Software Quality - 2

- Defect prevention methods
- Non-test defect removal methods
- Testing methods
- User-satisfaction methods
- Post-release quality control

Best in Class Quality Results - 1

- Quality measurements
- Defect prevention
- Defect and quality estimation automation
- Defect tracking automation
- Complexity analysis tools

Best in Class Quality Results - 2

- Test coverage analysis tools
- Formal inspections
- Formal testing by test specialists
- Formal quality assurance group
- Executive and managerial understanding of quality

FINAL WORD

- Reductions in total defect potentials using methods of defect prevention
- Improvements in cumulative removal efficiency levels

Project Management Approaches and High Software Quality - 1

- Use of automated project estimation methods
- Use of automated project planning methods
- Use of early and automated estimates of software defect potentials
- Use of early and automated estimates of software defect removal efficiency
- Formal risk-analysis

Project Management Approaches and High Software Quality - 2

- Provision of adequate time for pre-test inspections
- Historical quality data from similar projects available
- Milestone tracking automated and thorough
- Defect tracking automated and thorough
- Management focus concentrated on achieving excellent results

Project Management Approaches and Poor Software Quality

- Exact opposite of the project management approaches correlating with high software quality

SQA Group - 1

- Every company, which wants to establish a reputation for producing high quality software, must establish a Software Quality Assurance (SQA) Group within the company
- This groups must be funded properly and management must pay attention to the reports and presentations made by this group

SQA Group - 2

- The SQA group report directly to the higher management and not to the project management
- The personnel of the SQA group must work with the project management team, and vice versa to produce high quality software for the company – which is the ultimate goal

- The SQA group is needed to monitor the quality assurance-related activities in a company

SQA Group's Activities - 1

- Preparation of an SQA plan for a project
- Participation in the development of the project's software process description
- Review of software engineering activities to verify compliance with the defined software process

SQA Group's Activities - 2

- Audit of designed software work products to verify compliance with those defined as part of the software process

SQA Group's Activities - 3

- Ensure that deviations in software work and work products are documented and handled according to a documented procedure
- Record any noncompliance and reports to senior management

SQA Plan - 1

- Evaluations to be performed
- Audits and reviews to be performed
- Standards that are applicable to the project
- Procedures for error reporting and tracking

SQA Plan - 2

- Documents to be produced by the SQA group
- Amount of feedback provided to the software project team

Software Quality Personnel

- Unfortunately are under-paid
- Usually are let go first in times of crisis
- "Top-gun" SQA personnel and managers with proven track record are in high demand from companies that have active QA programs

Costs of Software Quality - 1

- Defects prevention costs
- User satisfaction optimization costs
- Data quality defect prevention costs
- Data quality defect removal costs
- Quality awareness/training costs
- Non-test defect removal costs
- Testing defect removal costs

Costs of Software Quality - 2

- Post-release customer support costs
- Litigation and damage award costs
- Quality savings from reduced scrap/rework
- Quality savings from reduced user downtime
- Quality value from reduced time-to-market intervals

COSTS OF SOFTWARE QUALITY - 3

- Quality value from enhanced competitiveness
- Quality value from enhanced employee morale
- Quality return on investment

ECONOMICS OF SOFTWARE QUALITY - 1

- High quality software applications have shorter development schedules than low quality applications because they do not get hung up in integration and testing due to excessive defect levels

Economics of Software Quality - 2

- High quality software applications have lower development and maintenance costs than low quality applications. This is because the cumulative costs of finding and fixing bugs is often the major cost driver for software projects

ECONOMICS OF SOFTWARE QUALITY - 3

- High quality software applications have better reliability levels and longer mean times to failure than low quality applications
- High quality commercial software packages have larger market shares than low quality commercial software packages

ECONOMICS OF SOFTWARE QUALITY - 4

- High quality software achieves better user-satisfaction ratings than low quality software
- High quality software projects score better on employee morale surveys than do low quality software projects

Economics of Software Quality - 5

- High quality software produced under contract or an outsource agreement has a much lower probability of ending up in court for breach of contract or malpractice litigation than low quality software
- High quality software benefits or augments the performance levels of users, while poor quality tends to degrade worker performance

ECONOMICS OF SOFTWARE QUALITY - 6

- Poor quality software can trigger truly massive unplanned expense levels. Denver airport example


Thanks!