SOFTWARE QUALITY ASSURANCE

Lecture 2

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
- What is Quality?
- What is Software Quality?
- What is a Defect?
- Quality Control, Quality Assurance, and Quality Engineering

Donate **blood**

to save lives

Lets Start ☺

Discussion

- What is Quality?
- Is Quality Necessary?
- Does Quality effects anything?


START

Introduction

- Traditional definition of quality:
  - fitness of purpose,
    - a quality product does exactly what the users want it to do.

Fitness of purpose

- For software products,
  - fitness of purpose:
    - satisfaction of the requirements specified in SRS document.

Fitness of purpose

- A satisfactory definition of quality for many products:
  - a car, a table fan, a food mixer, microwave oven, etc.
- But, not satisfactory for software products.

Introduction

- Consider a software product:
  - functionally correct,
    - i.e. performs all functions as specified in the SRS document,
  - but has an almost unusable user interface.
    - cannot be considered as a quality product.

Introduction

- Another example:
  - a product which does everything that users want.
  - but has an almost incomprehensible and unmaintainable code.

Modern view of quality

- Associates several quality factors with a software product :
  - Correctness
  - Reliability
  - Efficiency (includes efficiency of resource utilization)
  - Portability
  - Usability
  - Reusability
  - Maintainability

Quality assurance

- Basic premise of modern quality assurance:
  - if an organization's processes are good and are followed rigorously,
    - the products are bound to be of good quality.

Quality assurance

- All modern quality paradigms include:
  - guidance for recognizing, defining, analyzing, and improving the production process.

Quality control (QC)

- Quality control aims at correcting the causes of errors:
  - not just rejecting defective products.
- Statistical quality control
  - quality of the output of the process is inferred using statistical methods
  - instead of inspection or testing of all products

???

- The next breakthrough,
  - development of quality assurance principles

Quality Engineering???

What is Quality?

- Can you define quality?
- You must be thinking, what kind of question is that.
- It is very easy to define quality, but if you think really hard, it is not that easy to define quality
- Have you come with a definition?

Synonyms of Quality

- Excellence
- Superiority
- Class
- Eminence
- Value
- Worth

Antonym of Quality

- Inferiority

Marketability of Quality

- Everyone claims to manufacture / develop / sell / market “good” quality products / services
- You will never come across a person or company selling products or services as low or poor quality products, even when they are

Software Quality - 1

- Quality as it relates to all aspects of software (requirements / design / code / tests / documents / training)
- Difficult to define
  - Software quality is somewhat like the concept of beauty.
  - Each of us has a strong opinion about what constitutes beauty, and we recognize it when we see it.
  - But when asked to explain exactly why we regard an object as beautiful, it is hard to put the factors into words

Software Quality - 2

- Good software quality characteristics can be identified
- Bad or undesirable characteristics can also be identified

Software Quality factors

- **Six key factors**, which are considered as definitions of software quality, and we'll use them from now onwards

KEY FACTORS

- Low levels of defects when deployed, ideally approaching zero
- High reliability, or the capability of running without crashes or strange results
- A majority of clients with high user-satisfaction when surveyed

KEY FACTORS

- A structure that can minimize “bad fixes” or insertion of new defects during repairs
- Effective customer support when problems do occur
- Rapid repairs for defects, especially for high-severity defects

Beyond Absence of Defects

- Sense of beauty
- Sense of fitness for purpose
- Has well-formed requirements

WHY SOFTWARE QUALITY? - 1

- Reduces time to market for new products
- Enhances market share compared to direct competitors
- Minimizes “scrap and rework” expenses
- Attracts and keeps “top-gun” personnel
- Minimizes the risk of serious litigation

WHY SOFTWARE QUALITY? - 2

- Minimizes the risk of serious operating failures and delays
- Minimizes the risk of bankruptcy or business failures, which may be attributed directly to poor quality or poor software quality

How do you define quality now

Software Quality Assurance

- So the term software quality assurance would mean that the software guarantees high quality
- In this course, we'll learn the different processes, techniques, and activities, which enables us – the software professionals – to provide that guarantee to ourselves and our clients

Achieving Software Quality

- "For a software application to achieve high quality levels, it is necessary to begin upstream and ensure that intermediate deliverables and work products are also of high quality levels. This means that the entire process of software development must itself be focused on quality"
  - Capers Jones

What is a Software Defect?

- A software defect is an error, flaw, mistake, failure, or fault in software that prevents it from behaving as intended (e.g., producing an incorrect or unexpected result)
- Software defects are also known as software errors or software bugs

Effects of Software Defects - 1

- Bugs can have a wide variety of effects, with varying levels of inconvenience to the user of the software. Some bugs have only a subtle effect on the program's functionality, and may thus lie undetected for a long time. More serious bugs may cause the software to crash or freeze leading to a denial of service

Effects of Software Defects - 2

- Others qualify as security bugs and might for example enable a malicious user to bypass access controls in order to obtain unauthorized privileges

Effects of Software Defects - 3

- The results of bugs may be extremely serious
- In 1996, the European Space Agency's US $1 billion prototype Arian 5 rocket was destroyed less than a minute after launch, due a bug in the on-board guidance computer program

Effects of Software Defects - 4

- In June 1994, a Royal Air Force Chinook crashed into the Mull of Kintyre, killing 29 people. An investigation uncovered sufficient evidence to convince that it may have been caused by a software bug in the aircraft's engine control computer

Effects of Software Defects - 5

- In 2002, a study commissioned by the US Department of Commerce' National Institute of Standards and Technology concluded that software bugs are so prevalent and detrimental that they cost the US economy and estimated US $59 billion annually, or about 0.6 percent of the gross domestic product

Categories of Software Defects

- Errors of commission
- Errors of omission
- Errors of clarity and ambiguity
- Errors of speed or capacity

Errors of Commission

- Something wrong is done
- A classic example at the code level would be going through a loop one time too many or branching on the wrong address

Errors of Omission

- Something left out by accident
- For example, omitting a parentheses in nested expressions

Errors of Clarity and Ambiguity

- Different interpretations of the same statement
- This kind of error is common with all natural language requirements and specification documents and user manuals, too.

Errors of Speed and Capacity

- Application works, but not fast enough

Software Defect Origins

- Software defects can be found in any of the documents and work products including very serious ones in cost estimates and development plans
- However, there are seven major classes of software work products where defects have a strong probability of triggering some kind of request for warranty repair if they reach the field

Software Defect Origins

- Errors in Requirements
- Errors in Design
- Errors in Source code
- Errors in User Documentation
- Errors due to “Bad fixes”
- Errors in Data and Tables
- Errors in Test Cases

Defect Discovery

- Defects are discovered by developers & testers (usually) before release
- Defects are discovered by customers and users (usually) after release
- Defects discovered after release can be embarrassing for the development team

Software Defect Elimination Strategies

- Effective defect prevention
- High levels of defect removal efficiency
- Accurate defect prediction before the project begins
- Accurate defect tracking during development
- Useful quality measurements
- Ensuring high levels of user-satisfaction

Defect Prevention and Removal

- Both defect prevention and removal techniques are used by the "best-in-the-class" companies
- Defect prevention is very difficult to understand, study, and quantify. We'll talk about defect prevent in a later lecture
- Both non-test and testing defect removal techniques must be applied

Typical Defect Removal

- Inspections
  - Direct fault detection and removal
- Testing
  - Failure observation and fault removal

INSPECTIONS - 1

- Inspections are critical examinations of software artifacts by human inspectors aimed at discovering and fixing faults in the software systems

INSPECTIONS - 2

- Inspections are critical reading and analysis of software code or other software artifacts, such as designs, product specifications, test plans, etc

- Inspections are typically conducted by multiple human inspectors, through some coordination process. Multiple inspection phases or sessions may be used

INSPECTIONS - 3

- Faults are detected directly in inspection by human inspectors, either during their individual inspections or various types of group sessions

- Identified faults need to be removed as a result of the inspection process, and their removal also needs to be verified

INSPECTIONS - 4

- The inspection processes vary, but typically include some planning and follow-up activities in addition to the core inspection activity

- The formality and structure of inspections may vary, from very informal reviews and walkthroughs, to fairly formal variations of Fagan inspection, to correctness inspections approaching the rigor and formality of formal methods

Non-Test Defect Removal Methods

- Requirement inspections
- Design inspections
- Code inspections
- Test plan reviews
- Test-case inspections
- User documentation editing or reviews

Testing Defect Removal Methods

- Unit test by individual programmers
- New function testing
- Regression testing
- Performance testing
- Integration testing
- System testing

Defect Removal

- Not all defects are equal when it comes to removal
- Requirements errors, design problems, and "bad fixes" are particularly difficult

Defect Repair Rates - 1

- Defect repair rates increase with experience in application, language, inspections, structured design and coding methods
- Defect repair rates are higher for maintenance specialists than others during maintenance phase
- For coding errors, they correlate with comment density. IBM's study concluded that 18% comment density is ideal
- It also found, flow charts had no impact, but good error messages had great impact

Defect Seeding

- Willful insertion of errors into a software deliverable prior to a review, inspection, or testing activity
- It is the quickest way of determining defect removal efficiency
- Considered unpleasant by many

Defect Severity Levels

- Most software defect tracking systems include a multi-tier "severity level" scale

- For example,
  - Severity level 1: total failure of application
  - Severity level 2: failure of major function(s)
  - Severity level 3: minor problem
  - Severity level 4: cosmetic problem

Defect Tracking

- It is important to use an accurate and automated defect tracking system

- Defect tracking tools
  - Tracking defects by severity level and by origin
  - Routing defects to appropriate repair facility
  - Keeping records of duplicate defects
  - Invalid defects
  - Repair information against defects


Thanks!