Lecture 1: Introduction to Software Quality Assurance

Software Quality Assurance (INSE 6260/4-UU)
Winter 2016
Overview

- Course Outline
- Project
- Software Quality
- Software Quality Assurance
Course Outline

- Instructor: Dr. J. Bentahar
- Office: EV7.630
- Lectures: Tuesday, 17h45 – 20h15
- Office Hours: Friday, 10h30 – 12h00
- Phone: 848-2424 ext. 5382
- E-Mail: bentahar@ciise.concordia.ca
Course Outline

- Course Web:
  http://www.ciise.concordia.ca/~bentahar/inse6260.html
  - Lecture notes
  - Useful links
  - Useful information
Core Courses
- Total Quality Methodologies in Engineering
- Advanced Statistical Approaches to Quality
- Total Quality Project Management
- Seminar in Executive Communication

Quality Methodologies for Software
- Software Quality Assurance

Quality-Based Systems Eng.
Quality Assurance for System Eng.

Quality in logistics and SCM
- Quality Assurance in SCM

Software Systems Procurement
Software Quality Assurance:
- Software Quality (factors, models, life cycles, and standards)
- Quality assurance for quality models and specifications: verification, validation and testing

Objectives:
- To discover and learn various concepts and techniques related to software quality assurance
- To learn to apply these techniques
- To develop critical thinking skills
INSE 6260/4-UU

Software Quality Assurance

- Software Quality
  - Factors and Models
  - Metrics
  - Inspection
- Quality Assurance
  - Testing Techniques
  - Reachability Analysis
Textbooks for Reading

1) *Software Quality Assurance: From Theory to Implementation, 2012*

- This book covers several issues related to software quality assurance. Some important chapters are: software quality challenge, what is software quality, software quality factors, and software testing
Textbooks for Reading

- 2) Metrics and Models in Software Quality Engineering, 2010

  This book is a reference in software metrics. It covers a comprehensive breadth of measurement theory and software quality metrics.
Requirements and Grading

- One in-class midterm exam (closed book)
  - 28%
- One in-class final exam (closed book)
  - 40%
- One team project (2 members (No Exception), presentation + report)
  - 16% + 16% = 32%
## Important dates

<table>
<thead>
<tr>
<th>Event</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>Project proposal</td>
<td>February 02, 2016</td>
</tr>
<tr>
<td>Midterm exam</td>
<td>March 01, 2016</td>
</tr>
<tr>
<td>Project presentation</td>
<td>March 29, 2016</td>
</tr>
<tr>
<td>Final exam</td>
<td>TBS</td>
</tr>
<tr>
<td>Project report</td>
<td>April 12, 2016</td>
</tr>
</tbody>
</table>
Overview

- ✓ Course Outline
- Project
- Software Quality
- Software Quality Assurance
Project proposal: Crowd sourcing

- Deadline: February 02, 2016
- Team members
- Programming Language(s)
- Abstract about the architecture and main functionalities of the system
- Main references

The topic will be discussed next week in detail
Overview

- ✔ Course Outline
- ✔ Project
- Software Quality
- Software Quality Assurance
Definitions

- Software
- Software quality
- Software quality assurance
- Errors, faults, failures
- Cause of errors
What is Software?

IEEE definition, ISO 1997 sec. 3.11 and ISO/IEC 9000-3 SEC. 3.14

- Software is:
  - Computer programs (Code)
  - Procedures
  - Documentation
  - Data necessary for operating the software system
Software Quality

- SQ- IEEE definition
  - 1) The degree to which a system, component, or process meets specified requirements
  - 2) The degree to which a system, component, or process meets customer or user needs or expectations
SQ Pressman’s Definition

- SQ is defined as
  - Conformance to explicitly stated functional and performance requirements, explicitly documented development standards, and implicit characteristics that are expected of all professionally developed software
Quality Problem

Users

Decision maker
client

Request

Benefits

Decision maker
provider

Developers

Intermediate
Products

software

Quality
Quality Policies and Quality System

Expressed and implicit Requirements by the users

Quality policies

Quality system

Implements

Product (quality)
Overview

- ✓ Course Outline
- ✓ Software Quality
- Software Quality Assurance
- Software Development Life Cycle
Quality Software Assurance

- IEEE definition
  - Quality assurance is
    1. A planed and systematic pattern of all actions necessary to provide adequate confidence that an item or product conforms to established technical requirements
    2. A set of activities designed to evaluate the process by which the products are developed or manufactured
Software Quality Assurance (cont.)

- SQA – expanded definition
  - A systematic, planed set of actions necessary to provide adequate confidence that the software development process or the maintenance process of the software system product conforms to established functional technical requirements as well as with the managerial requirements of keeping the schedule and operating within budgetary confines
Quality Control versus SQA

- **Quality Assurance** is process oriented and focuses on defect *prevention*, while **quality control** is product oriented and focuses on defect *identification*.

- Quality Assurance (QA) is meant to minimize the costs of quality by introducing a variety of activities throughout the development process and maintenance process in order to prevent the causes of errors, detect them, and correct them in the early stages of the development.
  - As a result, quality assurance substantially reduces the rate of non-qualifying products
The Uniqueness of SQA

- High complexity
- Invisibility of software
- Production process

→ Need for SQA methodologies and tools
Why do we care about Quality?

<table>
<thead>
<tr>
<th>Development cost</th>
<th>Correction cost &amp; source</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specification:</td>
<td>Req. &amp; Specification:</td>
</tr>
<tr>
<td>Design:</td>
<td>56%</td>
</tr>
<tr>
<td>Coding:</td>
<td>Design:</td>
</tr>
<tr>
<td>V&amp;V (Testing):</td>
<td>24%</td>
</tr>
<tr>
<td>Maintenance:</td>
<td>Coding:</td>
</tr>
<tr>
<td></td>
<td>Other:</td>
</tr>
</tbody>
</table>

- Specification: 6%
- Design: 5%
- Coding: 7%
- V&V (Testing): 15%
- Maintenance: 67%
- Requirement & Specification: 56%
- Design: 24%
- Coding: 10%
- Other: 10%
Relative Cost of Error Correction
Systems Complexity

Systems Are Becoming More Complex

Estimated Onboard SLOC Growth

Curve implies SLOC doubles about every 4 years

The line fit is pegged at 27M SLOC because the projected SLOC sizes for 2010 through 2020 are unaffordable. The COCOMO II estimated costs to develop that much software are in excess of $10B.

Examples of Software Glitches

1998 Aegis missile cruiser

- USS Yorktown was crippled by the entry of a zero into a data field, causing the database to over flow and crash all LAN consoles and miniature remote terminal units.

- Protection against such bad data is a known design technique that was not used.

- ... the reported, corrective maintenance was not to fix the design, as would be expected, but to retrain the operators “to bypass a bad data field and change the value if such a problem occurs again”

Jet Propulsion Laboratory
Examples of Software Glitches

2007 Raptor (F-22) incident

- While attempting first overseas deployment to Japan, on 11 February 2007, a group of six F-22A Raptor flying from Hickam AFB, Hawaii experienced multiple computer crashes...
  - The computer failures included navigation (completely lost) and communication.
  - The problem seems to have arisen from the change in longitude from W179.99 degrees to E180 which occurs on the International Date Line.

- The fighters were able to return to Hawaii by following their tankers in good weather. The error was fixed within 48 hours and the F-22s continued their journey to Kadena.

F-22 Squadron Shot Down by the International Date Line

Mar 01, 2007 05:35 UTC by Defense Industry Daily staff

Aircraft software can be serious business. DID's F-22A Raptor FOCUS Article mentioned recent flight software problems that delayed the aircraft's first foreign deployment external link from Hickam AFB in Hawaii to Kadena AFB, Japan.

What we didn't mention at the time is how serious the problem was, and how dependent on computers modern aircraft - including military aircraft - have become.
Examples of Software Glitches

2012 B737 EFB-related Accident

- An onboard EFB was used to calculate the aircraft’s takeoff performance.

The commander omitted to enter the aircraft’s takeoff weight into the performance calculation software, which defaulted to the previous flight’s takeoff weight.

- The crew did not cross-check the data and incorrect speeds and thrust were calculated and subsequently used for the takeoff.

Aircraft Type and Registration: Boeing 737-33A, G-ZAPZ
Date & Time (UTC): 14 April 2012 at 10:08 hrs
Location: Chambéry Airport, France

Annual Safety Recommendations Review EASA, 2013
Distinction between Software Errors and Faults

- **Software Error**: section of the code that are incorrect as a result of grammatical, logical or other mistakes made by a system analyst, a programmer, or another member of the software development team.

- **Software Faults**: software errors that cause the incorrect functioning of the software during a specific application.
From Errors to Failures
Causes of Software Errors

- Client-developer communication failures
- Deviations from software requirements
- Logical design errors
- Coding errors
- Non compliance with documentation and coding instructions
- Shortcoming of the testing process
- Documentation errors
Cost of Quality

Cost of quality includes all costs incurred in the pursuit of quality or perform quality related work

Quality cost includes:

- Prevention cost:
  - quality planning
  - formal technical reviews
  - testing equipment
  - training

- Appraisal cost:
  - in-process and inter-process inspection
  - testing

- Failure cost:
  - rework, repair, and failure mode analysis
  - complaint resolution
  - product return and replacement
Three Views of Quality

- Product view:
  - functionality, correctness, performance, usability ...

- Process view:
  - efficiency, cost & schedule, failures (rework).

- Business view:
  - timeliness, customer satisfaction
  - does it sell?
  - producing the right product at the right time for the right price?
SQA Group

People involved in quality assurance activities:
Software engineers, project managers, customers, sales people, SQA group

Activities:
- apply technical methods and measures
- conduct formal technical review
- perform well-planned software testing
What Experts say about Quality?

Two dominant views on software quality:

1. Conformance and degree of satisfaction to defined specification.

2. The products or services capability to meet customer expectations – explicitly or implicitly stated.

Experts:

Philip B. Crosby
Edwards Deming
Armand V. Feigenbaum
Joseph M. Juran
Walter A. Shewhart
Kaoru Ishikawa
Quality is a subjectively identified notion by each individual and institution.

As this is not useful in software engineering, quality must be defined as “conformance to requirements”.

Nonconformance to requirements is the absence of quality, quality problems become nonconformance problems, and quality becomes definable.
W. Edwards Deming

- Translating future needs of the user into measurable characteristics

- Constantly changing based on “real world” competitors, solutions, technology, and price
Quality is based upon the customer’s actual experience with the product or service, measured against his or her requirements:

- stated or unstated
- conscious or merely sensed
- technically operational
- entirely subjective
- always representing a moving target
Quality defined according to standards (ISO, IEEE etc.) contain shortcomings, does not reflect constantly changing customer needs

- Narrowly interpreted, quality means quality of products

- Broadly interpreted, quality means quality of product, service, information, processes, people, systems etc.
Joseph M. Juran

- Quality can be:
  - Those product features which meet the need of customers and thereby provide product satisfaction
  - Freedom from deficiencies
  - Quality in terms of satisfying customer expectations or specifications is not usable as it is very hard to achieve
  - Quality is fitness for use
Quality from two perspectives:

- An objective reality independent of the existence of the customer
- The subjective perspective dependent on individual thoughts, feelings or senses as a result of the objective reality
References

- Chap 1, 2, 3 & 4 of “Software Quality Assurance” Daniel Galin