ILLINOIS INSTITUTE OF TECHNOLOGY

Transforming Lives. Inventing the Future.

www.iit.edu

SOFTWARE QUALITY MANAGEMENT

CSP587

Prof. Dennis Hood

Computer Science

Lesson Overview

- Quality Fundamentals
- Reading
  - Ch. 1 – Software Quality Fundamentals
- Objectives
  - Consider the relationship between quality and defects
  - Put quality into a management context
  - Discuss steps to be taken proactively (planned) which will have the effect of increasing the level of quality in a developed system
  - Analyze process, human, and system elements of quality management

Week 2
Quality Fundamentals

SQA Scope

- What is the scope of “concern”?
  - Defects can be internal
    - Analysis, design, and/or coding errors
  - or external
    - Users
    - Partners
    - The environment
- What then is the scope of the SQM’s job?
  - Establish and maintain a quality-focused organization
  - Minimize (eliminate?) the likelihood of system failures
  - Guide the creation of defect-free systems
  - Protect deployed systems against external threats

Software System Quality

- Software defined (ISO)
  - All or part of the programs, procedures, rules, and associated documentation of an information processing system
  - Computer programs, procedures, and possibly associated documentation and data pertaining to the operation of a computer system
- Software “problems” – errors, defects, and failures
  - Performance inconsistent with agreed upon requirements
  - “Inserted” during development (analysis, design, or coding); “appear” during testing or runtime
  - Can a defect exist but never be seen?
  - Can an error occur without a defect existing?

Seek First to Understand …

- Understanding the source of defects is the first step in preventing them
  - Defects are inserted into a system under development
  - Not intentionally, but rather in spite of efforts not to insert them
- Anticipation
  - Anticipating insertion opportunities is the first step in preventing them (proactive)
    - An umbrella should protect the development environment (similar to a clean room)
  - Anticipating the “location” of defects is the first step in “discovering” them (reactive)
    - Surgical care should be taken in removing them

Common Causes of Errors

- Problems with defining requirements
- Ineffective client-engineer communication
- Deviations from specifications / requirements
- Architecture / design errors
- Coding errors
- Non-compliance with processes / procedures
- Inadequate reviews / tests
- Documentation errors

Automated Error Detection

- Automation benefits from established rules and predictability
- Having concise definitions of the root causes of a set of common errors makes automated defect detection more feasible
- Is automated defect removal next?

Software Quality (ISO definition)

- The capability of a software product to satisfy stated and implied needs when used under specified conditions
- The degree to which a software product meets established requirements
  - however, quality depends upon the degree to which those established requirements accurately represent stakeholder needs, wants, and expectation

Requirements

- Functional requirements
  - The “what”
  - The basis for test plans
- Quality requirements
  - The “how”
  - The basis for expanded testing?
    - Stress testing, usability analysis, etc.
- Analysis and documentation
  - If you don’t know what it was supposed to do AND how it was supposed to do it, then how do you test it?

Quality Factors

- Operation factors
  - Correctness (of the output)
  - Reliability (of service)
  - Efficiency (of hardware resources)
  - Integrity (protecting the data from unauthorized access)
  - Usability (effort to achieve user proficiency)
- Revision factors
  - Maintainability (effort to manage incidents)
  - Flexibility (managing modifications, etc.)
  - Testability (ease of verifying adherence to requirements)
- Transition factors
  - Portability (effort required to support a new environment)
  - Reusability (ability to use components in new systems)
  - Interoperability (ability to interface with other systems)

Alternative Quality Factors

- Verifiability
  - Features that enable efficient verification
- Expandability
  - Support for scalability
- Safety
  - Meant to eliminate conditions “hazardous” to the user
- Manageability
  - Support for configuration and change management
- Survivability
  - Continuity of service

An SQA Architecture

- Pre-project components
- Project life-cycle components
- Infrastructure components
- Management components
- Standards, certification and assessment components
- Human components

Software Quality Assurance

- A set of activities that define and assess the adequacy of software processes to provide evidence that establishes confidence that the software processes are appropriate for and produce software products of suitable quality for their intended purposes
- A key attribute of SQA is the objectivity of the SQA function with respect to the project
- The SQA function may also be organizationally independent of the project – i.e., free from technical, managerial, and financial pressures from the project

Business Model Influence

- Purpose dictates priority
  - Does the level of quality mean the difference between life and death? (mission criticality)
  - Does it drive profit margin? (perceived value)
  - Are there legal ramifications? (check the contract)
- Environment matters
  - Size and makeup of the user community
  - Competition
  - Complexity / risk
  - Regulations
- Tradeoffs
  - Time to delivery vs level of quality
  - Ease of use / performance vs security
  - Can we afford to invest in the future?, can we afford not to?
  - DIY vs outsource