SE4831 Software Quality Assurance

Dr. Walter Schilling

Winter, 2011-2012

For the midterm exam, you may bring one 8.5 x 11 inch sheet of paper with notes.

1. Lecture #1 (Introduction)
   (a) Quantify the impact of software quality problems as they currently exist.
   (b) Recognize and discuss the complex nature of modern software failure.
   (c) Explain Why Because Analysis and how it can be used to reach the root causes for a software problem
   (d) Recognize and discuss the complex nature of modern software failure
   (e) Explain the importance of organizational culture on quality
2. Lecture #2 (Organizing for Quality)
   (a) Define quality program.
   (b) Define software product.
   (c) Define software process.
   (d) Define requirement
   (e) Explain three relations that the developers may have with the customer.
   (f) Explain the difference between validation and verification.
   (g) Differentiate between the two major models of SQA
   (h) Explain what infrastructure is necessary for quality software development
   (i) Explain the difference between the internal and external view of quality.
   (j) Explain how software volatility can indicate the need for software requalification.
   (k) List the five levels of maturity from the CMMI model and explain how quality is impacted at each level.
   (l) Explain the role of independence as it relates to a quality program.
3. Lecture #4 (Software Quality Planning)
   (a) List the important aspects of a Software Quality Assurance Plan
   (b) List the key aspects of an IEEE 730 SQA Plan
   (c) Explain the concept of certification as it applies to software standards.
   (d) Justify appropriate Quality Assurance Practices given the domain and scope of a project
   (e) Construct a software quality assurance plan which is in conformance with IEEE-730.
4. Lecture #5, 6, 8 (Software Inspections)
   (a) Compare and contrast software inspections and walkthroughs.
   (b) item Explain how a software inspection can be used as a quality gate.
   (c) List the elements of a peer review.
   (d) List the roles for each participant in a software inspection and define their scope.
   (e) Draw a flowchart listing the steps for a software inspection and describe the activities that occur in each phase.
   (f) Explain how checklists can be used to improve the effectiveness of a review process.
   (g) Explain how generic checklists can yield reduced inspection effectiveness.
   (h) List the problems identified by the National Software Quality Experiment.
   (i) Critique inspection performance based on quantifiable metrics to identify potential problems.

(j) Explain the problem with using bug counts as the sole measure of review effectiveness.

(k) Explain the concept of capture-recapture experimental methods.

(l) Explain how capture-recapture methods can be used to assess the effectiveness of formal inspections.

(m) Using capture-recapture methods, estimate the remaining defects within a software artifact.

(n) Explain the concept of fault injection.

(o) Explain how fault injection can be combined with capture-recapture methods to assess review effectiveness.

(p) Perform a formal inspection on a software artifact using capture-recapture to assess the effectiveness of the review.

5. Lecture 9 (Software Quality Techniques - Pareto Principle)

(a) Explain how to construct a pareto chart.

(b) Explain how the pareto principle can be used during software development.

(c) Explain how institutional data and the pareto principle can be used to yield better inspection performance.

6. Lecture 10 (Release Management)

(a) Express the ramifications of stopping testing too soon or continuing testing too long

(b) Justify why it is appropriate to stop testing on a software development project