

# Software Quality Attributes

Jim Brosseau

Clarrus Consulting Group Inc.

# Overview

- What usually happens
- What are they, and why do I care?
- A more disciplined approach

# Achieving Software Quality

- Test for Errors
  - “does it execute without breaking?”
- Review for Defects
  - “does it meet the specs?”
- Engineer in Quality
  - “is my design reliable?”

# Typical Approaches

- Ignore them altogether
  - How do you know you have a quality product?
- Reuse the same attributes as the last project
  - Dangerous: safety, litigation issues
- Define them ambiguously
  - “the system shall be *User Friendly*”...

# The Project Triangle

- Scope, time, cost
  - Pick two, you can't have all 3!
- Quality is often neglected
  - Not defined up front
  - Implicitly degraded throughout project
  - We are still 'testing for errors' for the majority of our quality landscape

# Quality Attributes Landscape

**Availability:** Is it available when and where I need to use it?

**Efficiency:** How few system resources does it consume?

**Flexibility:** How easy is it to add new capabilities?

**Installability:** How easy is it to correctly install the product.

**Integrity:** Does it protect against unauthorized access, data loss?

**Interoperability:** How easily does it interconnect with other systems?

**Maintainability:** How easy it is to correct defects or make changes?

**Portability:** Can it be made to work on other platforms?

**Reliability:** How long does it run before experiencing a failure?

**Reusability:** How easily can we use components in other systems?

**Robustness:** How well does it respond to unanticipated conditions?

**Safety:** How well does it protect against injury or damage?

**Testability:** Can I verify that it was implemented correctly?

**Usability:** How easy is it for people to learn or to use?

# Alternative Taxonomies...

- Correctness
- Efficiency
- Expandability
- Flexibility
- Integrity
- Interoperability
- Maintainability
- Manageability
- Portability
- Usability
- Reliability
- Reusability
- Safety
- Survivability
- Verifiability

***Hewlett-Packard: FLURPS+***

# Key Considerations

- Can't satisfy all of them
  - Prioritize!
- Opportunity for Dev/Test teams
  - We're all Stakeholders!
- Break down the problem
  - Divide and conquer!
- Analogous to a discipline for project metrics: GQM, coined by Vic Basili

# Developing Quality Attributes

- Break down the task into manageable steps
  - Each is straightforward – no 'leaps of faith'
- Drive this in a collaborative session
- Capture the rationales for each decision along the way
  
- Spreadsheet and whitepaper provided as softcopy
- Great reference:
  - Michael S. Deutsch, Ronald R. Willis: Software Quality Engineering – A Total Technical and Management Approach. Prentice-Hall, 1988

# Summary

- A straightforward approach, simple steps
  - Allows us to quantify the quality expectations of the product
- Up-front definition of quality minimizes likelihood of implicit quality sacrifices

# Questions?