Quality and Quality Assurance

- Elements of Quality
  - definitions of quality
  - the economics of quality
  - pro-active multi-faceted approaches
- Quality Assurance Processes
  - roles of Quality Assurance
  - Quality Assurance vs. testing
  - Process Assurance
  - engineering Quality Assurance activities

Elements of Quality

- Customer-facing characteristics
  - functionality (usability, integration, power)
  - correctness (accuracy, reliability)
  - performance, scalability, robustness
  - manageability, flexibility
- Developer-facing characteristics
  - maintainability (modularity, testability, read-ability, simplicity)
  - supportability (diagnose-ability, service-ability)
  - extensibility (generality, portability)

Un-Quality Costs Money

- Finding bugs is expensive
  - more testers and more time
- Fixing bugs is more expensive
  - bug reports, code changes, additional testing
- Shipping bugs is even more expensive
  - costs of diagnosing user problems
  - costs of delivering work-arounds, patches
  - reduced user productivity, increased TCO
- Preventing bugs saves time and money

Defect Amplification

where found	where introduced		
	requirements	architecture design	construction
requirements	1x		
design	3x	1x	
construction	5-10x	10x	1x
system test	10x	15x	10x
post-ship	10-100x	25-100x	10-25x

Estimated Effectiveness

Process	Minimum	Average	Maximum
design reviews	25%	35%	40%
code reviews	20%	25%	30%
personal desk checking	20%	40%	60%
unit testing	15%	30%	50%
integration/system testing	25%	25%	40%
regression test suites	15%	25%	30%
Combined Total Effectiveness	**74%**	**90%**	**97%**

Testing vs. Quality Assurance

- many people equate Q/A with testing
  - testing is a way to measure product quality
  - discovering defects can assist their elimination
  - it is not an effective/efficient way to get quality
- some testing is best done by a Q/A group
  - unit testing is best done by developers
  - whole system, acceptance testing is different
- but Q/A is much more than testing
  - find and eliminate all sources of non-quality

Typical Roles for S/W Q/A

- Test Group
  - system acceptance, performance, etc
  - methodology experts and/or test execution
- Process Assurance
  - maintain the rules, monitor compliance
- Product Data Collection and Reporting
  - providing management with objective data
- Customer Advocate
  - counterbalance to development and sales

Q/A Process Assurance

- Engineering is responsible for quality
  - their processes create the work products
- Quality Assurance monitors process to ...
  - capture work products and metrics
  - assess engineering process compliance
  - report on product/process status
  - drive process assessment and improvement
- Engineering and Q/A jointly work to ...
  - understand process weaknesses
  - define process improvements

Engineering Quality Assurance

- Many process steps try to ensure quality
  - ensure decisions based on good information
  - ensure use of best practices
  - find/eliminate errors as quickly as possible
- Engineering performs most of these steps
  - engineering requirements validation
  - reviews (architecture, design, code)
  - unit test case design, development, execution
  - configuration management
  - bug management

Requirements Reviews

- Ensuring we are building the right thing
- user-level requirements
  - clear and well justified, widely agreed to
  - traceable and prioritized
  - relatively complete and stable
  - do we believe we can satisfy them?
- validate component-level requirements
  - reasonable, complete, consistent, testable
  - do they add up to the user-level requirements

Architectural Reviews

- Review architecture prior to design
- Is it capable of meeting requirements?
  - embraces all applicable standards
  - no performance or robustness issues
- Will it be practical to build & support?
  - all components well specified, look doable
  - reasonable use of off-the-shelf technology
  - good modularity, well abstracted interfaces
- Is there anything here we'll regret later?

Design Reviews

- Review Design prior to implementation
- Is it clear how to build this component?
  - no significant open questions
- Is the design reasonable?
  - it will satisfy all component requirements
  - complete, correct, and relatively simple
  - no major concerns about how it will work
- Is it clear how to test this component?
  - we know how to build the test cases
  - running them will give us good confidence

Code Reviews

- Review Code prior to integration/testing
- Does this code implement the design?
  - implements all specified functionality
  - appropriately handles all reasonable cases
- Is this code obviously correct
  - unobviousness often hides incorrectness
- Will this code be supportable?
  - reasonably structured and commented
- Does it conform to applicable standards?

Configuration Management

- Ensuring a reproducible product
- define a standard build environment
  - system, compiler, libraries, other tools
  - package it for use by all developers
- create/maintain build scripts
  - to automate construction of the product
- create/maintain version control files
  - keeping track of all changes (what and why)
  - enabling reconstruction of any version.

Engineering - Unit Testing

- Ensuring that delivered components work
- specify component test cases
  - to validate that component works properly
    - usually functionality and error handling
  - based on requirements and design
- review component test plan
  - to ensure its adequacy
- implement component test cases
  - usually under a standard testing harness
- regularly run component test cases

Engineering - Bug Management

- Analysis - what is the actual problem
  - bug, documentation, misunderstanding, ...
  - if bug, what component, what consequences
- Triage - sort bugs (must-fix, should-fix, defer)
- develop work-arounds, patches, fixes
- update regression tests – to catch in future
- update defect tracking data base
- root-cause analysis – prevent future errors

Team Exercise

- Define “quality” attributes for your product
  - based on phase 1 research and interviews
  - other than “correctness”
  - particularly important to its success
- Refine those definitions to be measurable
  - suggesting a means for testing them
- Suggest a means of achieving them?
  - what can you do to ensure them
  - at what steps of your process
- Be prepared to discuss

For Next Lecture (design reviews)

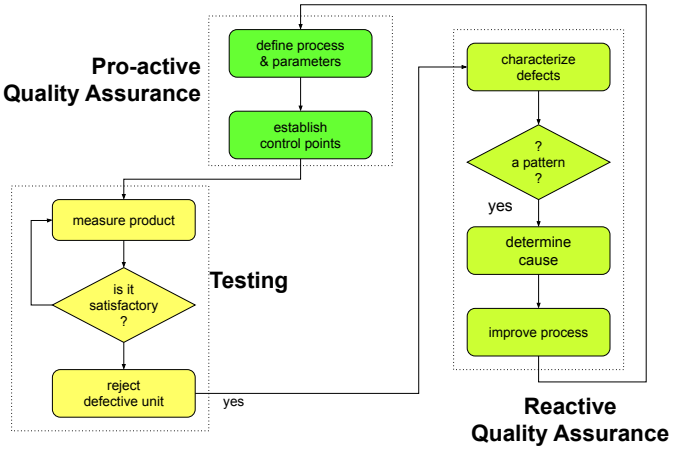
- Wiegers: Inspections
  - principles and detailed techniques
- Wiegers: Seven Deadly Sins
  - how reviews most commonly go wrong
- McConnell 21.4
  - walk-throughs and other less formal processes

Backup Slides

The need for Quality Assurance

- software is too complex to just work
  - simple building & testing is failure assurance
- it is sure that many mistakes will be made
  - the question is when/how we'll find them
- experience shows that sooner is cheaper
  - this is widely recognized and accepted
- which brings us to the question of how
  - the answer is through proactive processes
  - we call these processes Quality Assurance

Traditional Manufacturing Q/A



Anecdotal Process Improvement

- study a defective product
  - to understand why it came out defective
- figure out where our process failed
  - what steps allowed this defect to happen
- update our process
  - to prevent such mistakes in the future
- this approach works
  - but some major problems remain un-fixed
  - some fixes don't wind up making a difference

Statistical Quality Assurance

- gather data on all defects
  - including severity and root-cause analysis
- apply statistical techniques to this data
  - identify causes of the most/worst problems
  - focus on finding and fixing these causes
- change engineering processes
  - to avoid making such mistakes in the future
- change process metrics/control points
  - to permit us to better manage the process