

Software Process for QA

Planning testing and analysis
 throughout a software
 development cycle
 Basic Processes and Alternatives
 Process Issues

ICS 224 4/13/00 1

Goals and Constraints

- **Organization goals & priorities differ**
 - time-to-market, feature list vs. reliability, robustness; priorities for a pacemaker will not be the same as for a spreadsheet
- **Process constrains quality measures**
 - Example: Reliability is measurable only late in the process, so we may measure an earlier indicator (e.g., fault density)

ICS 224 4/13/00 2

Designing a process for quality

- **Process and quality goals are intertwined**
 - process can be designed to improve quality
 - quality assurance must be crafted around process constraints, especially, schedule and resources
- **Development process embodies quality goals and a model of risks**

ICS 224 4/13/00 3

Waterfall Model (example)

from Ghezzi et al, Fundamentals of Software Engineering, 1991

Each passage from phase to phase is marked by completion of a document that governs the following phase

ICS 224 4/13/00 4

Detail of a Waterfall Stage

- Goal is an output document consistent with the input document; an "error" is an inconsistency
- Phase is complete when document is accepted
- Each phase has specific methods

ICS 224 4/13/00 5

Royce's Waterfall Model

W. Royce, "Managing the Development of Large Software Systems." Proc. IEEE WESCON, Aug 1970.

ICS 224 4/13/00 6

Quality process: Cleanroom

Mills, Dyer & Linger, "Cleanroom Software Engineering." IEEE Software 4(5), Sep 1987.

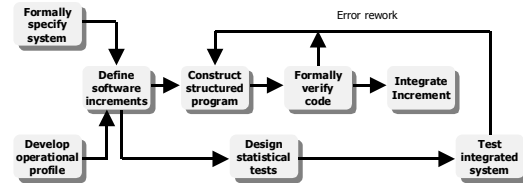
- "Cleanroom" software development process
 - Formal verification at unit level; no unit testing
 - Independent statistical testing with simple accept/reject outcome
- Embodied model of quality
 - Avoiding "fault injection"
 - strong bias toward waterfall model of development
 - Goal is reliability (only)

ICS 224

4/13/00

7

The Cleanroom Process Model



ICS 224

4/13/00

8

SRET* Process

J. Musa, Software Reliability Engineering: More Reliable Software, Faster Development and Testing. 1998.

Define "necessary" reliability

Develop Operational Profile

Prepare test cases

Interpret
Execute failure tests

Feasibility/Rqmts	Architecture and Design	Implementation	Component & System Test
-------------------	-------------------------	----------------	-------------------------

ICS 224

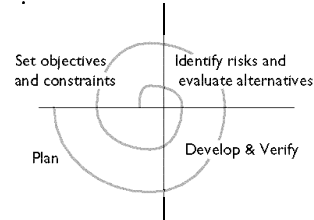
4/13/00

9

Spiral model

B Boehm, "A Spiral Model of Software Development and Enhancement." IEEE Computer, May 1976.

Risk-based process



ICS 224

4/13/00

10

Comparing Waterfall to Spiral

- Assumptions about quality risks
 - Naive waterfall: Detail design and coding errors
 - earlier errors are caught late, and at great cost
 - Spiral: Identifiable aspects of design
 - those identified and evaluated with a prototype
 - may be performance, usability, cost, etc. as well as dependability
- Different opportunities to measure and enhance quality

ICS 224

4/13/00

11

Front-end Quality Activities

- Quality cannot be "added on"
- Sequential models (waterfall)
 - Risk assessment, robust specifications
 - Specify and design for testability
 - Acceptance test plan and earlier measures
- Risk-based models (spiral)
 - All of the above (for each wind), plus choosing incremental builds for early assessment

ICS 224

4/13/00

12

Organizational Issues

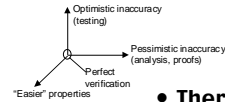
- Risk and reward system must avoid “perverse incentive”
 - Example: reporting fault avoidance and identification must not be risky
- Lines of responsibility influence behavior
 - Developer responsibility vs. independent test
 - Who is the boss of the tester?
 - Can problems be “thrown over the wall”?

ICS 224

4/13/00

13

Complementary Techniques



- There are no silver bullets
 - Different techniques depending on available artifacts
 - Different techniques depending on properties to be ascertained
 - Different techniques depending on acceptable cost and degree of assurance

ICS 224

4/13/00

14

Automation

- Automation is sometimes necessary
 - For some automated static analyses
 - To accurately check large sets of test results
- ... and often useful
 - For cost-effective regression testing
 - To monitor coverage, generate some tests automatically
- But some techniques are practical even without tools
 - Inspections of all artifacts, compilation of checklists, functional test case creation

ICS 224

4/13/00

15

Visibility

- How is status measured against goals, throughout development
 - A general process issue, applies also to schedule, cost, etc.
- Challenge is early visibility
 - Progress against QA plans
 - Early assurances and predictors (e.g., of testability)

ICS 224

4/13/00

16

Designing a Feedback Mechanism

- We lack good data about the nature and sources of faults
 - Information for fault avoidance, early removal, and better measurement
- Feedback can be built into the process
 - example: SRET process includes identification of how fault occurred, how it could be avoided, and how it could be identified

ICS 224

4/13/00

17

Example process

Requirements Elicitation	Requirements Specification	Architectural Design	Detail Design	◆◆◆
✓ Identify qualities	✓ Validate specifications	✓ Architectural design inspection	✓ Design inspections	
✓ Acceptance test planning	✓ System test planning	✓ Integration & unit test planning	✓ Generate oracles	
	✓ Create functional tests	✓ Automated architectural design analysis	✓ Generate black-box test cases	
			✓ Automated design analyses	

ICS 224

4/13/00

18

Example Process (cont.)

◆◆◆	Detail Design	Unit Coding	Integration & Delivery	Maintenance
	✓ Design inspections	✓ Code inspections	✓ Integration test execution	✓ Regression test execution
	✓ Generate oracles	✓ Create scaffolding	✓ System test execution	✓ Revise regression test suite
	✓ Unit test planning	✓ Unit test execution	✓ Acceptance test execution	
	✓ Automated design analyses	✓ Automated code analyses	✓ Deliver regression test suite	
		✓ Coverage analysis		

ICS 224

4/13/00

19

Analysis and Testing is Not a Phase

- **“Not a phase, but a lifestyle”**
 - Quality assessment and improvement activities must be spread through the whole development cycle
- **Planning is essential**
 - To achieve early and continuous *visibility*
 - To choose appropriate techniques at each stage
 - To build a verifiable product
 - To coordinate complementary analysis and testing techniques
- **Analyses and Tests are part of the work product**
 - Developed together with specifications, code, documentation, ...
 - Preserved and maintained for regression analysis and test
- **Design for testability/verifiability**
 - Both process and product structures can be adjusted to facilitate quality assessment and improvement

ICS 224

4/13/00

20