

▶ POLITECNICO DI MILANO

Dipartimento di  
Elettronica e Informazione

Session 10

# Integration & Testing

Emanuele Della Valle

<http://home.dei.polimi.it/dellavalle>

- This slides are largely based on Prof. John Musser class notes on “Principles of Software Project Management”
- Original slides are available at <http://www.projectreference.com/>
- Reuse and republish permission was granted

# Today

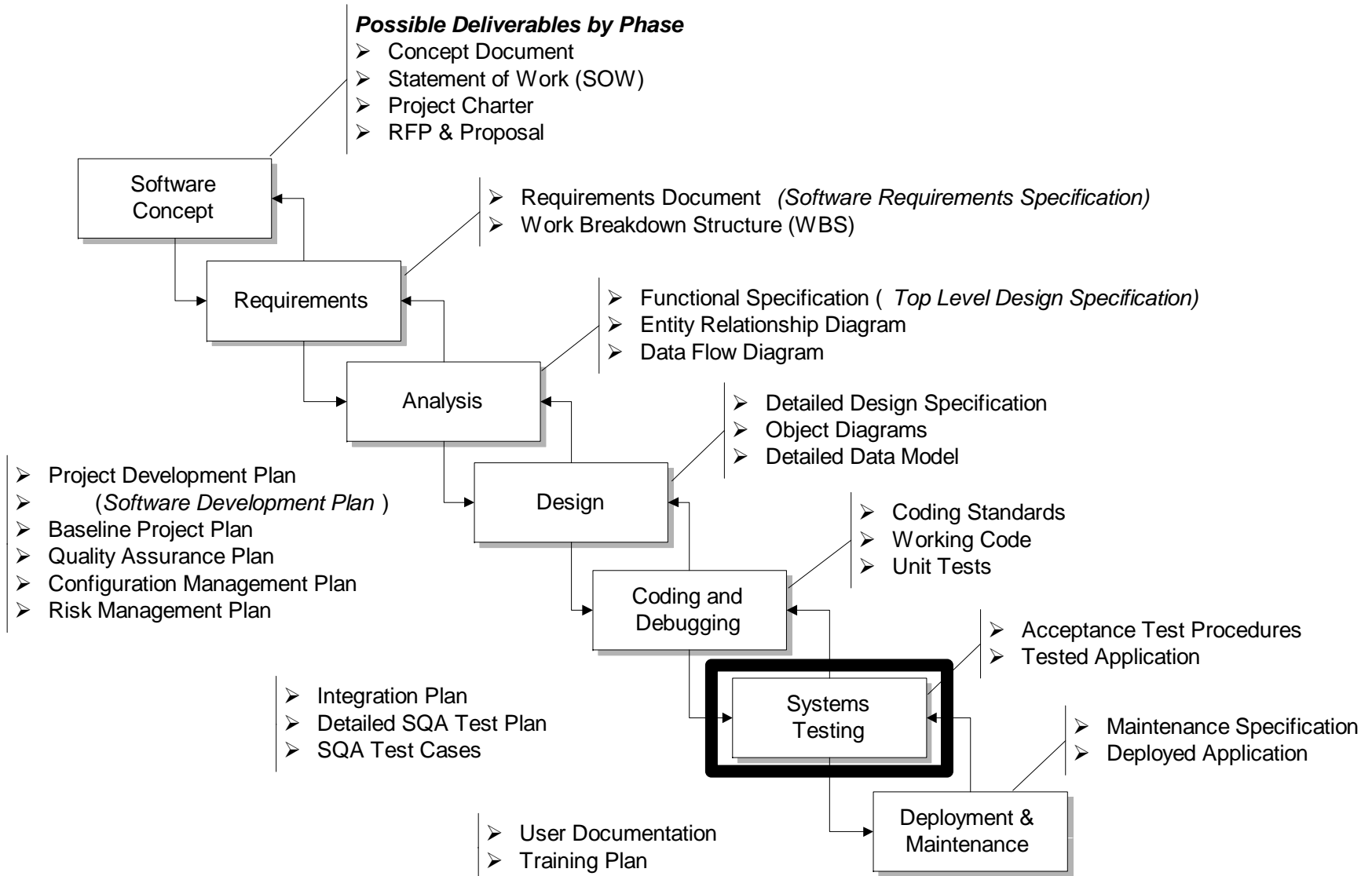
3

- Software Quality Assurance
- Integration
- Test planning
- Types of testing
- Test metrics
- Test tools

- Project Control
  - Planning
  - Measuring
  - Evaluating
  - Acting

- BCWS
- BCWP
  - Earned value
- ACWP
- Variances
  - CV, SV
- Ratios
  - SPI, CPI, CR
- Benefits
  - Consistency, forecasting, early warning

# Deliverables by Phase – Testing!

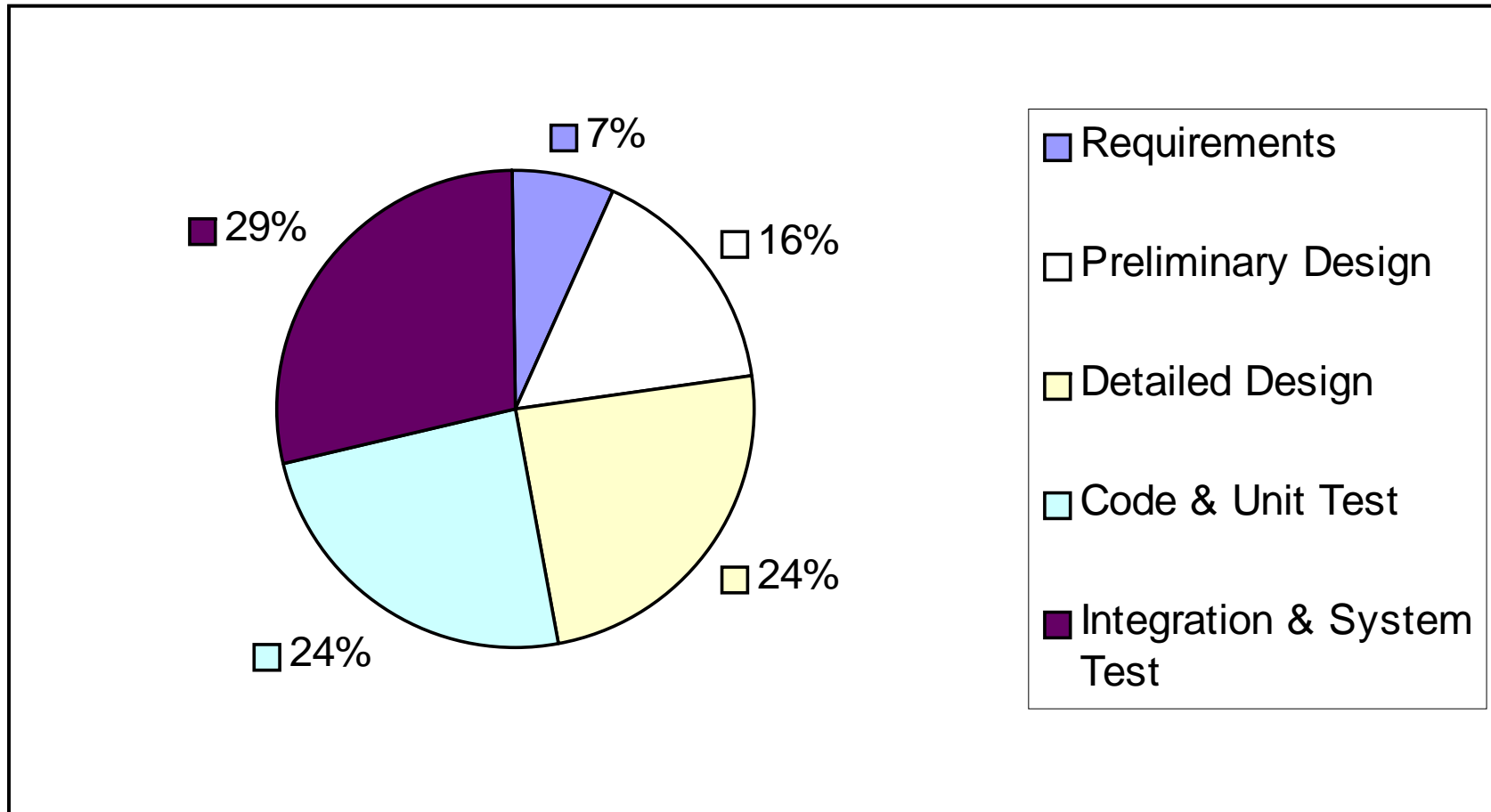


## If 99.9% Were Good Enough

7

- 9,703 checks would be deducted from the wrong bank accounts each hour
- 27,800 pieces of mail would be lost per hour
- 3,000,000 incorrect drug prescriptions per year
- 8,605 commercial aircraft takeoffs would annually result in crashes

Futrell, Shafer, Shafer, "Quality Software Project Management", 2002



- Development/Integration/Testing
  - Most common place for schedule & activity overlap
- Sometimes Integration/Testing thought of as one phase
- Progressively aggregates functionality
- QA team works in parallel with dev. team

- Top Down
  - Core or overarching system(s) implemented 1st
  - Combined into minimal “shell” system
  - “Stubs” are used to fill-out incomplete sections
    - Eventually replaced by actual modules
- Bottom Up
  - Starts with individual modules and builds-up
  - Individual units (after unit testing) are combined into sub-systems
  - Sub-systems are combined into the whole

- Who does integration testing?
  - Can be either development and/or QA team
- Staffing and budget are at peak
- “Crunch mode”
- Issues
  - Pressure
  - Delivery date nears
  - Unexpected failures (bugs)
  - Motivation issues
  - User acceptance conflicts

- V & V
- Validation
  - Are we building the right product?
- Verification
  - Are we building the product right?
  - Testing
  - Inspection
  - Static analysis

- QA or SQA (Software Quality Assurance)
- Good QA comes from good process
- When does SQA begin?
  - During requirements
- A CMM Level 2 function
- QA is your best window into the project

## Test Plans (SQAP)

14

- Software Quality Assurance Plan
  - Should be complete near end of requirements
- See example
  - [http://www.cio.energy.gov/documents/csr\\_sqa\\_plan.pdf](http://www.cio.energy.gov/documents/csr_sqa_plan.pdf)
    - Even use the IEEE 730 standard
      - <http://ieeexplore.ieee.org/iel4/5838/15568/00720573.pdf>

- Standard sections
  - Purpose
  - Reference documents
  - Management
  - Documentation
  - Standards, practices, conventions, metrics
    - Quality measures
    - Testing practices

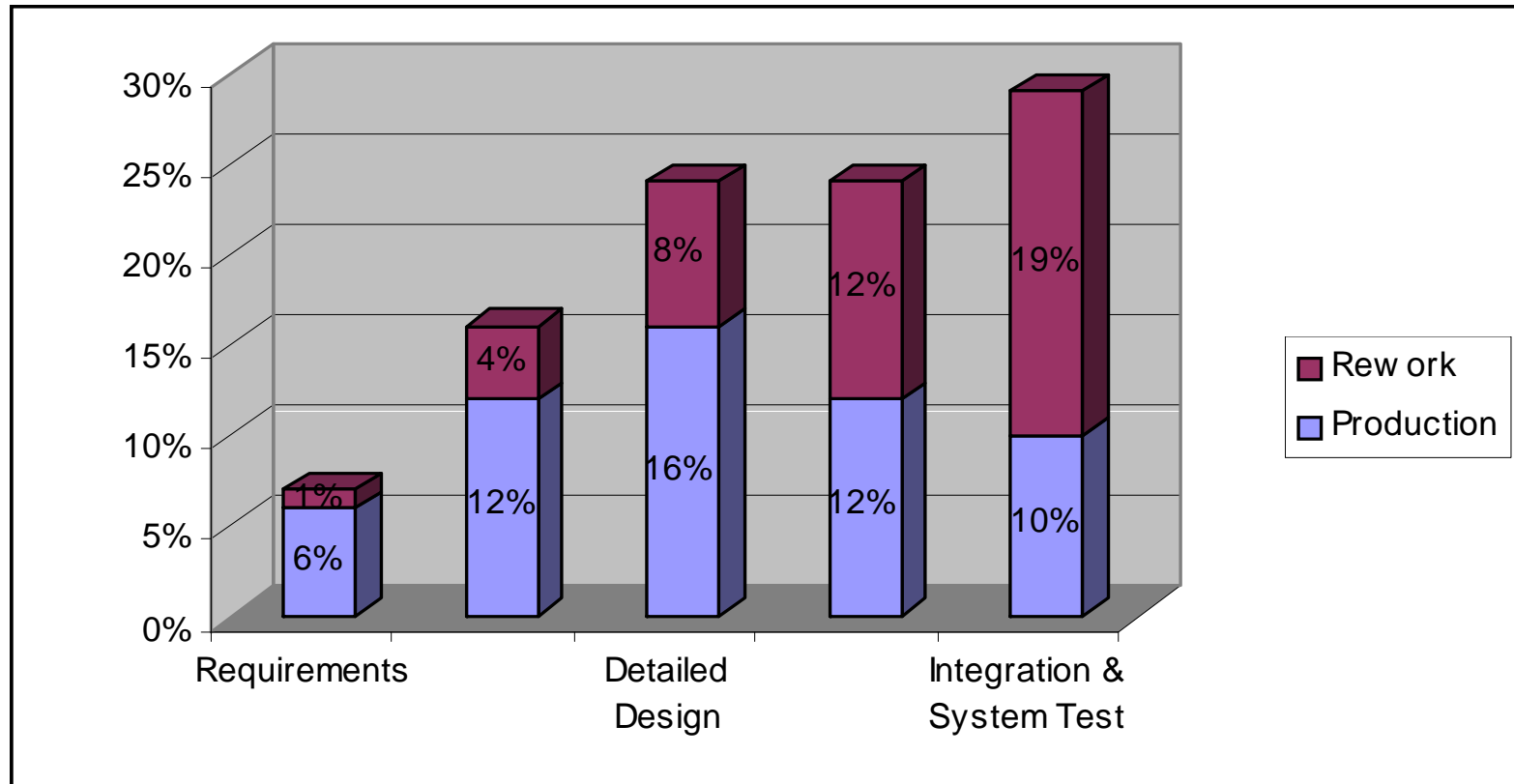
- Standard sections continued
  - Reviews and Audits
    - Process and specific reviews
      - Requirements Review (SRR)
      - Test Plan Review
      - Code reviews
      - Post-mortem review
  - Risk Management
    - Tie-in QA to overall risk mgmt. Plan
  - Problem Reporting and Corrective Action
  - Tools, Techniques, Methodologies
  - Records Collection and Retention

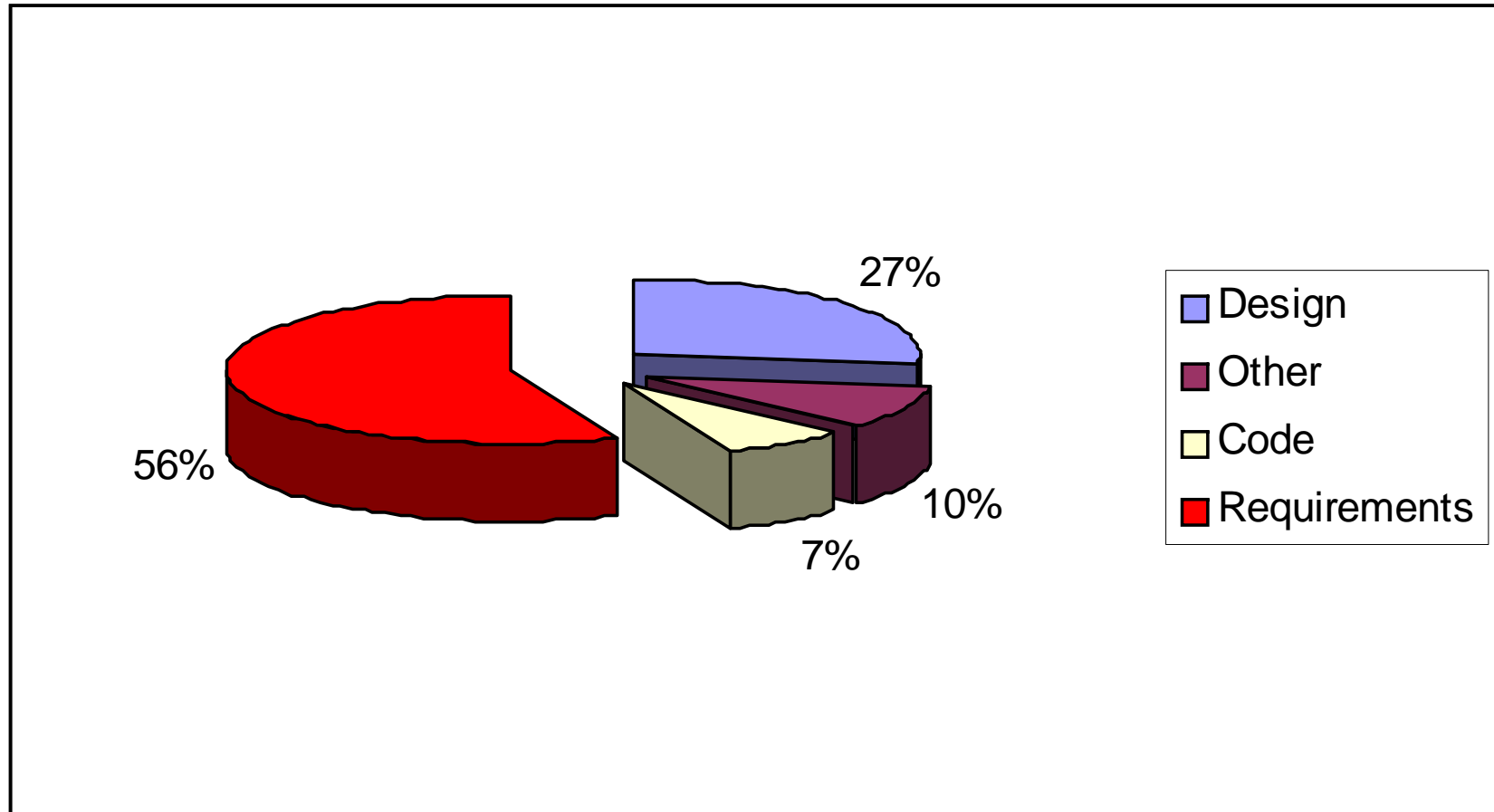
- Traceability
  - Ability to track relationship between work products
  - Ex: how well do requirements/design/test cases match
  
- Formal Reviews
  - Conducted at the end of each lifecycle phase
  - SRR, CDR, etc.

- Exercising computer program with predetermined inputs
- Comparing the actual results against the expected results
- Testing is a form of sampling
- Cannot absolutely prove absence of defects
- All software has bugs. Period.
- Testing is not debugging.

- Key elements of a test plan
- May include scripts, data, checklists
- May map to a Requirements Coverage Matrix
  - A traceability tool

- Software equivalent of "scrap" in manufacturing







## Project Testing Flow

23

- Unit Testing
- Integration Testing
- System Testing
- User Acceptance Testing

- Functional Testing
- Program is a “black-box”
  - Not concerned with how it works but what it does
  - Focus on inputs & outputs
- Test cases are based on SRS (specs)

- Accounts for the structure of the program
- Coverage
  - Statements executed
  - Paths followed through the code

- a.k.a. Module Testing
- Type of white-box testing
  - Sometimes treated black-box
- Who does Unit Testing?
  - Developers
  - Unit tests are written in code
    - Same language as the module
    - a.k.a. “Test drivers”
- When do Unit Testing?
  - Ongoing during development
  - As individual modules are completed

- Individual tests can be grouped
  - “Test Suites”
  
- JUnit
  - Part of the XP methodology
  - “Test-first programming”

- Testing interfaces between components
- First step after Unit Testing
- Components may work alone but fail when put together
- Defect may exist in one module but manifest in another
- Black-box tests

- Testing the complete system
- A type of black-box testing

- Last milestone in testing phase
- Ultimate customer test & sign-off
- Sometimes synonymous with beta tests
- Customer is satisfied software meets their requirements
- Based on “Acceptance Criteria”
  - Conditions the software must meet for customer to accept the system
  - Ideally defined before contract is signed
  - Use quantifiable, measurable conditions

- Re-running of tests after fixes or changes are made to software or the environment
- EX: QA finds defect, developer fixes, QA runs regression test to verify
- Automated tools very helpful for this

- Testing against other “platforms”
  - Ex: Testing against multiple browsers
  - Does it work under Netscape/IE, Windows/Mac
  
- Test repeatability and self-evidence
  - “Acid test” from the California Gold Rush
    - See [http://en.wikipedia.org/wiki/Acid\\_test\\_\(gold\)](http://en.wikipedia.org/wiki/Acid_test_(gold))
  - Acid tests for browsers compatibility to Web Standards
    - [http://en.wikipedia.org/wiki/Category:Acid\\_tests](http://en.wikipedia.org/wiki/Category:Acid_tests)
  - Be aware of
    - Quick and obvious vs. complete and fair

- Alpha 1st, Beta 2nd
- Testing by users outside the organization
  - Typically done by users
- Alpha release
  - Given to very limited user set
  - Product is not feature-complete
- During later portions of test phase
- Beta release
  - Customer testing and evaluation
  - Most important feature
  - Preferably after software stabilizes

- Value of Beta Testing
  - Testing in the real world
  - Getting a software assessment
  - Marketing
  - Augmenting you staff
- Do not determine features based on it
  - Too late!
- Beta testers must be “recruited”
  - From: Existing base, marketing, tech support, site
- Requires the role of “Beta Manager”
- All this must be scheduled by PM

- Release Candidate (RC)
  - To be sent to manufacturing if testing successful
- Release to Manufacturing (RTM)
  - Production release formally sent to manufacturing
- Aim for a “stabilization period” before each of these milestones
  - Team focus on quality, integration, stability

- Two meanings
  1. Set of step-by-step instructions intended to lead test personnel through tests
    - List of all actions and expected responses
  2. Automated test script (program)

- Reviews
  - Most artifacts can be reviewed
  - Proposal, contract, schedule, requirements, code, data model, test plans
  
- Peer Reviews
  - Methodical examination of software work products by peers to identify defects and necessary changes
  - Goal: remove defects early and efficiently
  - Planned by PM, performed in meetings, documented
  - CMM Level 3 activity

- Human testers = inefficient
- Pros
  - Lowers overall cost of testing
  - Tools can run unattended
  - Tools run through 'suites' faster than people
  - Great for regression and compatibility tests
  - Tests create a body of knowledge
  - Can reduce QA staff size
- Cons
  - But not everything can be automated
  - Learning curve or expertise in tools
  - Cost of high-end tools \$5-80K (low-end are still cheap)

- Capture & Playback
- Coverage Analysis
- Performance Testing
- Test Case Management

- Push system beyond capacity limits
- Often done via automated scripts
  - By the QA team
  - Near end of functional tests
- Can show
  - Hidden functional issues
  - Maximum system capacity
  - Unacceptable data or service loss
  - Determine if “Performance Requirements” met
    - Remember, these are part of “non-functional” requirements

- Metrics
  - Minimal acceptable response time
  - Minimal acceptable number of concurrent users
  - Minimal acceptable downtime
  
- Vendors: High-End
  - Segue
  - Mercury
  - Empirix

- Source: Athens Consulting Group

<b>Bad</b>	<b>Good</b>
Must support 500 users	Must support 500 simultaneous users
10 second response time	[Average Maximum 90th percentile] response time must be X seconds
Must handle 1M hits per day	Must handle peak load of 28 page requests per second

- Installation Testing
  - Very important if not a Web-based system
  - Can lead to high support costs and customer dissatisfaction
  
- Usability Testing
  - Verification of user satisfaction
    - Navigability
    - User-friendliness
    - Ability to accomplish primary tasks

- Pareto Analysis
  - The 80-20 rule
    - 80% of defects from 20% of code
  - Identifying the problem modules
- Phase Containment
  - Testing at the end of each phase
  - Prevent problems moving phase-to-phase
- Burn-in
  - Allowing system to run “longer” period of time
  - Variation of stress testing

- “Code Freeze”
  - When developers stop writing new code and only do bug fixes
  - Occurs at a varying point in integration/testing
  
- Tester-to-Coder Ratio
  - It depends
  - Often 1:3 or 1:4
  - QA staff size grows: QA Mgr and/or lead early

- When do you stop?
- Rarely are all defects “closed” by release
- Shoot for all Critical/High/Medium defects
- Often, occurs when time runs out
- Final Sign-off (see also UAT)
  - By: customers, engineering, product mgmt.,

- Load: Max. acceptable response time, min. # of simultaneous users
- Disaster: Max. allowable downtime
- Compatibility: Min/Max. browsers & OS's supported
- Usability: Min. approval rating from focus groups
- Functional: Requirements coverage; 100% pass rate for automated test suites

- These are very important to the PM
- Number of outstanding defects
  - Ranked by severity
    - Critical, High, Medium, Low
    - Showstoppers
- Opened vs. closed

- Get tools to do this for you
  - Bugzilla - <http://www.bugzilla.org/>
  - TestTrack Pro - <http://www.seapine.com/ttpro.html>
  - Rational ClearCase - <http://www-01.ibm.com/software/awdtools/clearcase/>
  - Some good ones are free or low-cost
- Make sure all necessary team members have access (meaning nearly all)
- Have regular 'defect review meetings'
  - Can be weekly early in test, daily in crunch
- Who can enter defects into the tracking system?
  - Lots of people: QA staff, developers, analysts, managers, (sometimes) users, PM

- Fields
  - State: open, closed, pending
  - Date created, updated, closed
  - Description of problem
  - Release/version number
  - Person submitting
  - Priority: low, medium, high, critical
  - Comments: by QA, developer, other

- Open Rates
  - How many new bugs over a period of time
- Close Rates
  - How many closed over that same period
  - Ex: 10 bugs/day
- Change Rate
  - Number of times the same issue updated
- Fix Failed Counts
  - Fixes that didn't really fix (still open)
  - One measure of "vibration" in project

- Microsoft Study
  - 10-20/KLOC during test
  - 0.5/KLOC after release

- You need to test somewhere. Where?
- Typically separate hardware/network environment(s)

- Development
- QA
- Staging (optional)
- Production

- Typical environments
  - Development
    - Where programmers work
    - Unit tests happen here
  - Test
    - For integration, system, and regression testing
  - Stage
    - For burn-in and load testing
  - Production
    - Final deployment environment(s)

- Unique factors
  - Distributed (N-tiers, can be many)
  - Very high availability needs
  - Uses public network (Internet)
  - Large number of platforms (browsers + OS)
- 5 causes of most site failures
  - Internal network performance
  - External network performance
  - Hardware performance
  - Unforeseeable traffic spikes
  - Web application performance

- Commercial Tools: Load Test & Site Management
  - Mercury Interactive
    - SiteScope, SiteSeer
  - Segue
- Commercial Subscription Services
  - Keynote Systems
- Monitoring Tools
  - [http://www.networkworld.com/reviews/2000/webmgt2result.jsp?\\_tablename=webmgt2](http://www.networkworld.com/reviews/2000/webmgt2result.jsp?_tablename=webmgt2)
- Availability: More “Nines” = More \$’s
  - Must balance QA & availability costs vs. benefits

- QA Manager
  - Hires QA team; creates test plans; selects tools; manages team
  - Salary: \$50-80K/yr, \$50-100/hr
- Test Developer/Test Engineer
  - Performs functional tests; develops automated scripts
  - Salary: \$35-70K/yr, \$40-100/hr
- System Administrator
  - Supports QA functions but not official QA team member
- Copy Editor/Documentation Writer
  - Supports QA; also not part of official team

- Your final Project schedule due 19.6.2009
  - Add resources and dependencies to your plan
  - Add durations and costs
  - Remember homework 4 is the most important homework I assigned you
  
- Please Submit electronically
  - TO: [emanuele.dellavalle@polimi.it](mailto:emanuele.dellavalle@polimi.it)
  - SUBJECT: [P&MSP09] final copy of <project-code> plan
    - e.g., [P&MSP09] final copy of 01 plan
  - Attachment a zip named as <project-code><student-ID> + <project name>
    - e.g., 01-740648-736424-SHOP HIVE Project.zip
  - Content of the zip
    - **Homework 4 !!!**
    - Homework 1-2-3 (I appreciate)

## Optional Readings

60

- McConnell: 16 “Project Recovery”
- Schwalbe: 16 “Closing”

# Questions?

61