Quality Assurance: Testing and Balancing

1 March 2011
CMPT420 / COMM350
Sean Ho
Trinity Western University

Further reading:
SoftwareQATest.com

TRINITY WESTERN UNIVERSITY

Quiz ch22

- At what milestone in production would a full team of testers be needed?
- What format/layout should a test plan use?
- Name 3 out of the 4 common bug definitions (types of bugs) noted in the text.
- What's the difference between the Severity and the Priority of a bug?
- Who is responsible for closing a bug?

Quiz ch22: answers #1-3

- At what milestone in production would a full team of testers be needed?
  - Alpha
- What format/layout should a test plan use?
  - Checklist or pass/fail
- Name 3 out of the 4 common bug definitions (types of bugs) noted in the text.
  - Crash, critical, minor, feature request

Quiz ch22: answers #4-5

- What's the difference between the Severity and the Priority of a bug?
  - Severity: crash/critical/minor/feature.
    Impact on gameplay or user experience.
  - Priority: order in which bug will be addressed;
    e.g., several bugs with same severity
- Who is responsible for closing a bug?
  - QA analyst or tester
  - (Not the developer or the one who fixed the bug!)

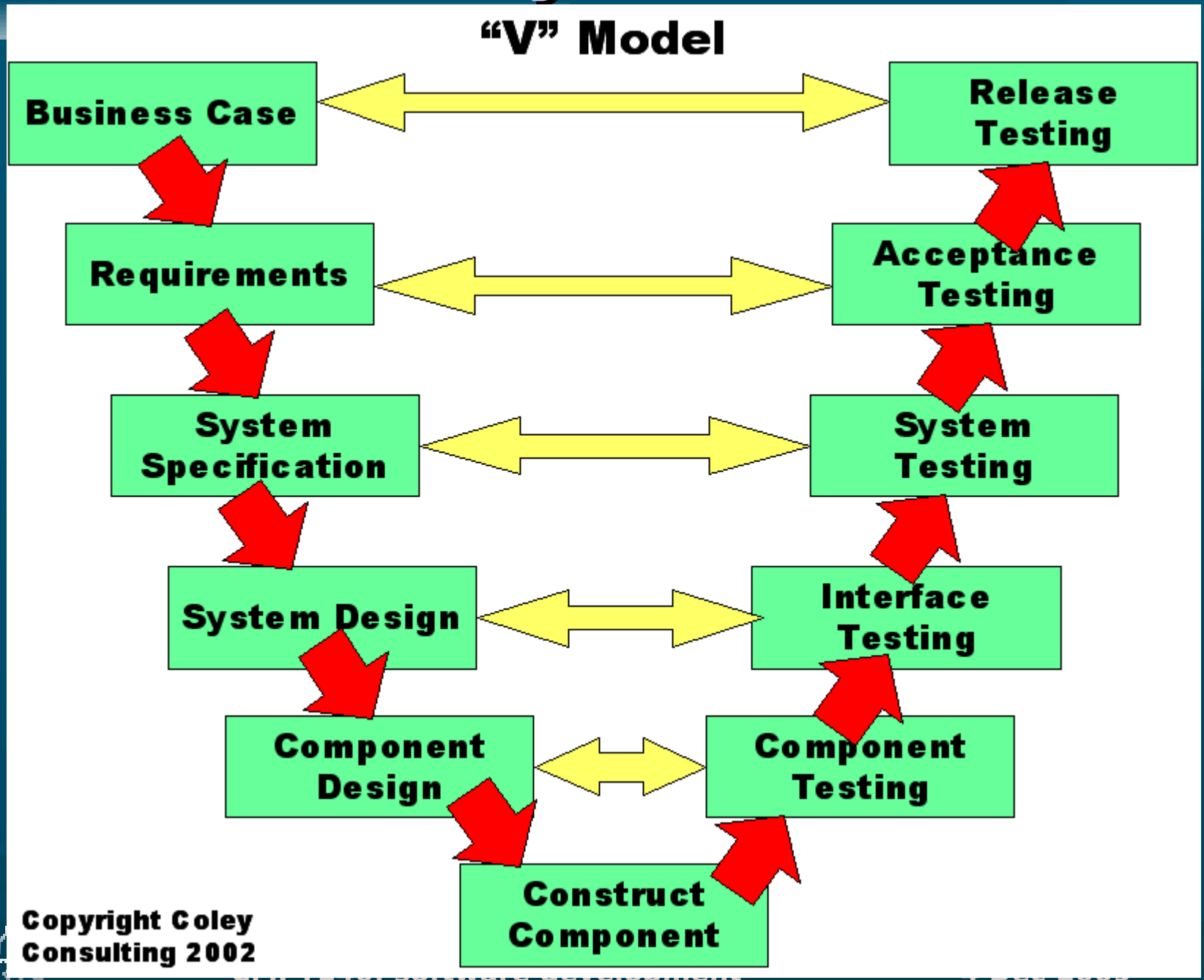
What's on for today

- QA is integral to the production process
- Approaches to testing:
  - Bottom-up vs. top-down
  - Black-box vs. white-box
  - Workflow vs. edge
  - Automated vs. manual
- Milestones in QA
- Submitting bug reports
  - Traffic flow of tickets

Quality assurance

- QA is not just squashing bugs! QA is:
- Satisfying requirements of the stakeholders
  - Producers, game studio ($$), target market
  - Servant leadership: are our clients happy?
- Delivering what we promised
  - Compare against specification/design docs
  - Compare against mission stmt, concept pitch
- QA happens at every stage of production
  - Prevention over cure

V model (Coley)



Testing reflects requirements

- Functionality: pot lifts when we click on it
- Performance: pot lifts within 50ms
- Usability: Easy to learn? Easy to remember? Efficient to use? Frequency/severity of errors? Subjective satisfaction of users?

- Also: security, scalability, fault-tolerance, accessibility, internationalization, compatibility / integration, … and more!
- Not knowing what to test reflects not knowing what the requirements are

Approaches to testing: 1

- Bottom-up testing: start with unit tests on each chunk of code (method, class, etc.)
  - Then integration testing of component intxns
  - Then system testing of the whole application
  - Exhaustive test coverage is difficult, tedious
- Top-down testing: start with end-user play
  - Components not used by end-user might not get tested ... but they may not be too relevant!
  - Multiple bugs may mask each others' effects

Approaches to testing: 2

- Black-box testing: without knowledge of the internal workings/code
  - e.g., play testing
    (by devs, by beta testers, or by target market)
  - e.g., monkey testing
    (random/haphazard clicking)
- White-box testing: “Use the source, Luke!”
  - Basis path testing: follow the flow of the program
  - Use a debugger to track objects, refs, memory
  - Document constants, magic numbers, etc.

Approaches to testing: 3

- Workflow (end-to-end, user-focused) testing:
  - Put yourself in user's shoes: download, install, run, and use it the way a user typically would
  - Best is to get a tester from target market
  - User might heavily use only 10% of your app (but which 10%?)
- Edge (boundary, stress) testing:
  - Try to break it, deliberately do strange / counter-intuitive things
  - What is counter-intuitive to you might not be so foreign to your end-users!

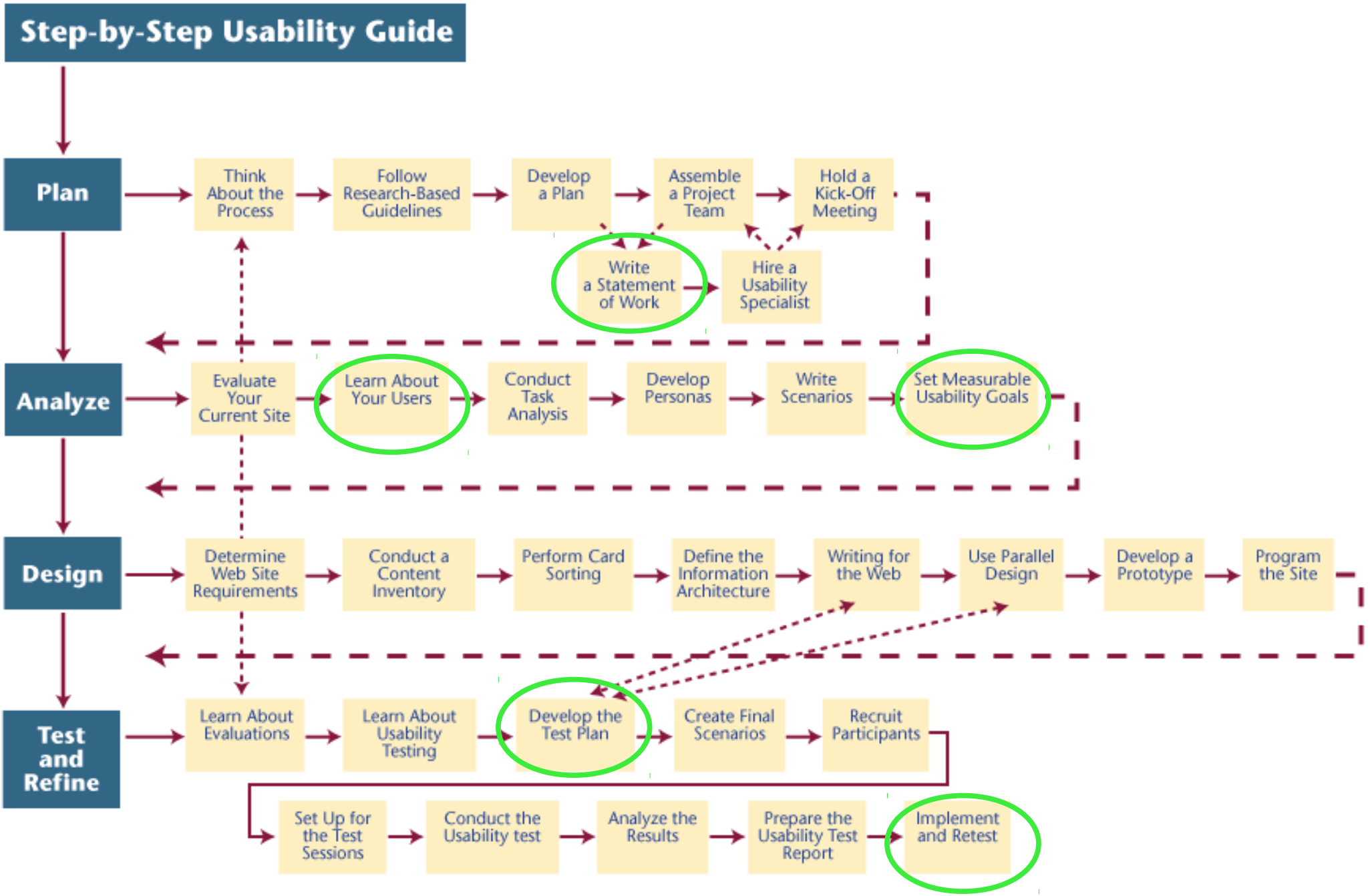
Approaches to testing: 4

- Automated (regression) testing:
  - Test cases organized into test suites
  - Fixtures setup toy environment for test cases
  - Dashboard shows status of nightly builds
  - Often used for lower levels (unit tests, etc.)
- Manual (human) testing:
  - Writing exhaustive automated test cases is just as hard as writing the code in the first place!
  - Usability, look-and-feel, user satisfaction are human intangibles

Milestones in QA

- Pre-alpha: internal white-box testing is ongoing and concurrent with development
- Alpha: send to internal black-box testing
  - Test team of non-developers
- Beta: send to external testers
  - Feature freeze; focus on usability testing
- Release candidate: ready for release
  - Acceptance testing: approved by stakeholders
- Gold master / RTM: release to public / sale
- Production / live: in daily use by clients

Example QA plan



How-to submit a bug report

- Refer to “Policies” and “Testing Procedures” on Trac wiki for details
- In the description, include:
  - Version you used (e.g., 0.1.6r234)
    - And OS (Windows 7, MacOS, etc.)
  - Severity of bug (crash, critical, minor, request)
  - Minimal sequence of steps to produce the bug
  - What behaviour you expected, and
  - What actually happened

Traffic flow of tickets

- Tester submits / creates new ticket
- QA analyst checks ticket for completeness, checks for duplicates, retitles as needed, and assigns to appropriate team lead
- Team lead may bounce it to another team, or assign priority and assign it to a member
- Team member handles bug and assigns back to QA analyst
- QA analyst coordinates with tester and closes ticket, or sends it back to team lead

Closing tickets

- Tickets may be closed as:
  - FIXED: tester and QA analyst are happy
  - NEEDSINFO: incomplete ticket submission (e.g., needs more info on user config, OS, etc.)
  - INVALID: not actually a bug (e.g., user misconfiguration)
  - WONTFIX: low priority, acceptable risk
  - DUPLICATE: same as another submitted ticket
  - WORKSFORME (Could Not Reproduce): may need more info from tester

QA and balancing

- QA is ensuring the product / service satisfies the requirements of stakeholders and delivers what was promised
- Usability and playability are important:
  - Playing games is voluntary
  - Competition in the game market is fierce
  - Good game play will make or break your game
- → Game balancing is a vital part of the QA process …