Test Estimation

Seeing the Future of Your Test Effort

RBCS

TIME TESTED.
TESTING IMPROVED.

www.RBCS-US.com

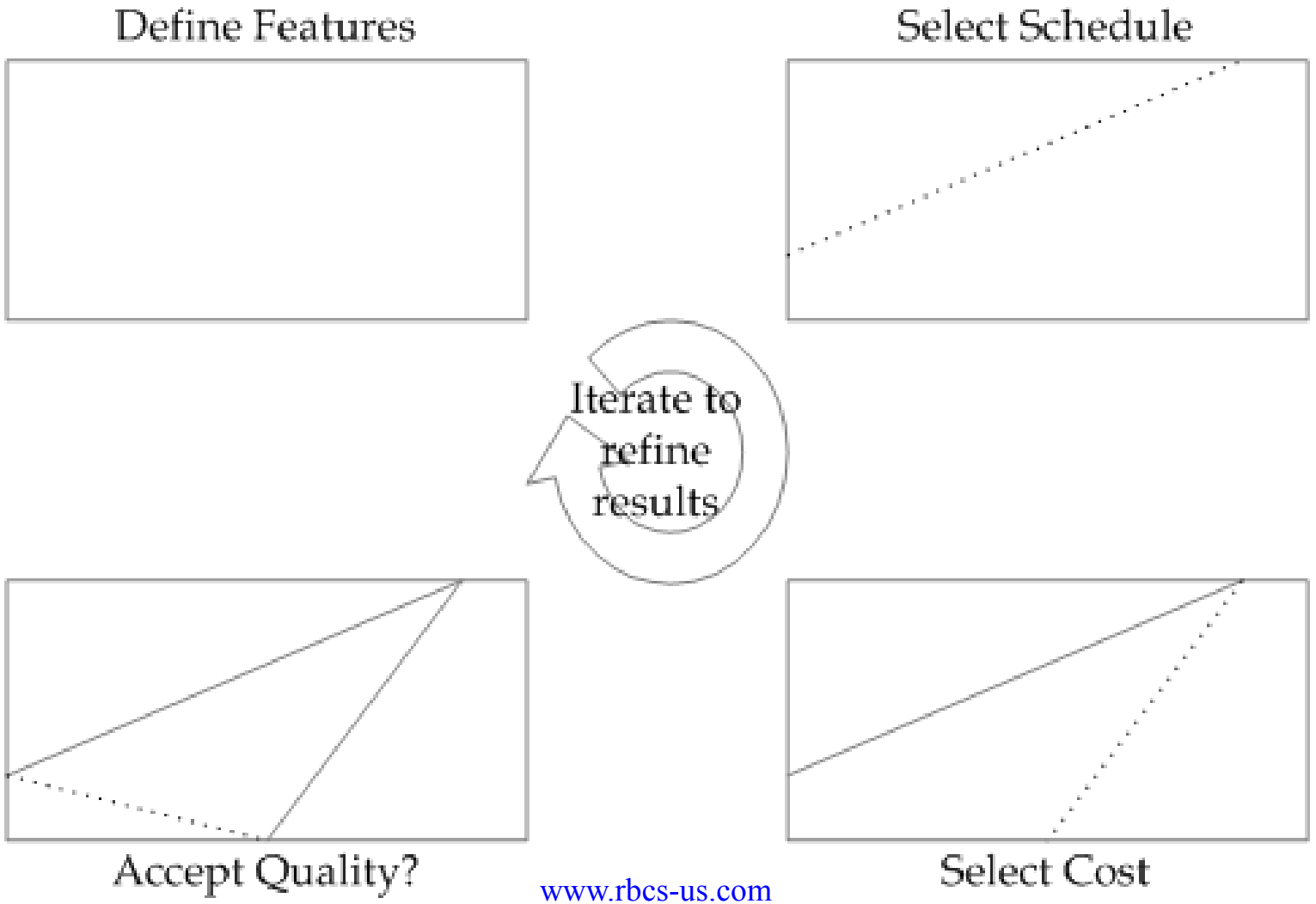
How Long Will Testing Take

- What makes an estimate a *good* one?
  - Accurately predicts and guides the project's future
  - Realistic: All tasks included, accurately estimated, risks understood and mitigated
  - Actionable: Clear ownership by committed individual contributors, assigned resources, known dependencies
- Estimation process
  - Ask experts and owners
  - Consult metrics and industry averages
  - Negotiate with managers and stakeholders

Manager and Stakeholder Expectations

- ☑ Cover the test basis completely, regardless of cost and schedule implications
- ☑ Cover the test basis as completely as possible within schedule constraints (cost not an issue)
- ☑ Cover the test basis as completely as possible within schedule and cost constraints
- ✘ Break the iron triangle (ignore interaction of schedule, budget, quality, features)

Iron Box and Triangle



Developing a Work-Breakdown-Structure

- What are the major stages of a testing subproject?
  - Planning
  - Staffing (if applicable)
  - Test environment acquisition and configuration
  - Test development
  - Test execution
- Break down to discrete tasks within each stage
- What test suites are required for the critical risks?
  - E.g., functionality, performance, error handling, etc.
- For each suite, what tasks are required?
- Tasks should be short in duration (e.g., a few days)
  - "Inch-pebbles" as well as "milestones"

Tapping the Team's Wisdom for Estimation

Delphic Oracle Technique

1. Gather estimates from each team member
2. Ask high and low estimators to explain estimates
3. Repeat twice, then use average

Three-Point Technique

1. Have team estimate best-case, worst-case, and expected-case
2. Use expected-case
3. Variances between best and worst useful to gauge estimate accuracy

These techniques can be combined (which is called the *Wideband technique*) and also used with other sources of estimation like historical project data, expert advice, and rules of thumb

Understanding Dependencies

Assign Dependencies

1. Identify tasks with no predecessors
2. Identify tasks dependent only on previously-identified tasks
3. Repeat step two until dependencies all identified

- Small projects: Can use project management tools (e.g., MS Project)
- Medium to large projects: Use index cards or sticky notes and whiteboard

Analyze Critical Paths

- A set of dependent tasks where delay in any task delays the project end
- Near-critical paths exist where significant delays will delay project end
- What affects phase entry and exit criteria?
- During test execution: How many test passes, releases, and cycles?
- External dependencies: a frequent cause of delay

Assigning Resources

People

- Test engineers and test technicians, contractors and employees, outside test resources
- Using less-skilled people increases task effort, duration
- People's skill with a given task or tool determines estimate accuracy

Test environments

- Hardware, software, networks, facilities, etc.
- Especially important to include expensive or long-lead-time items like large servers, test tools, lab space

Test tools

- Custom
- Commercial off-the-shelf

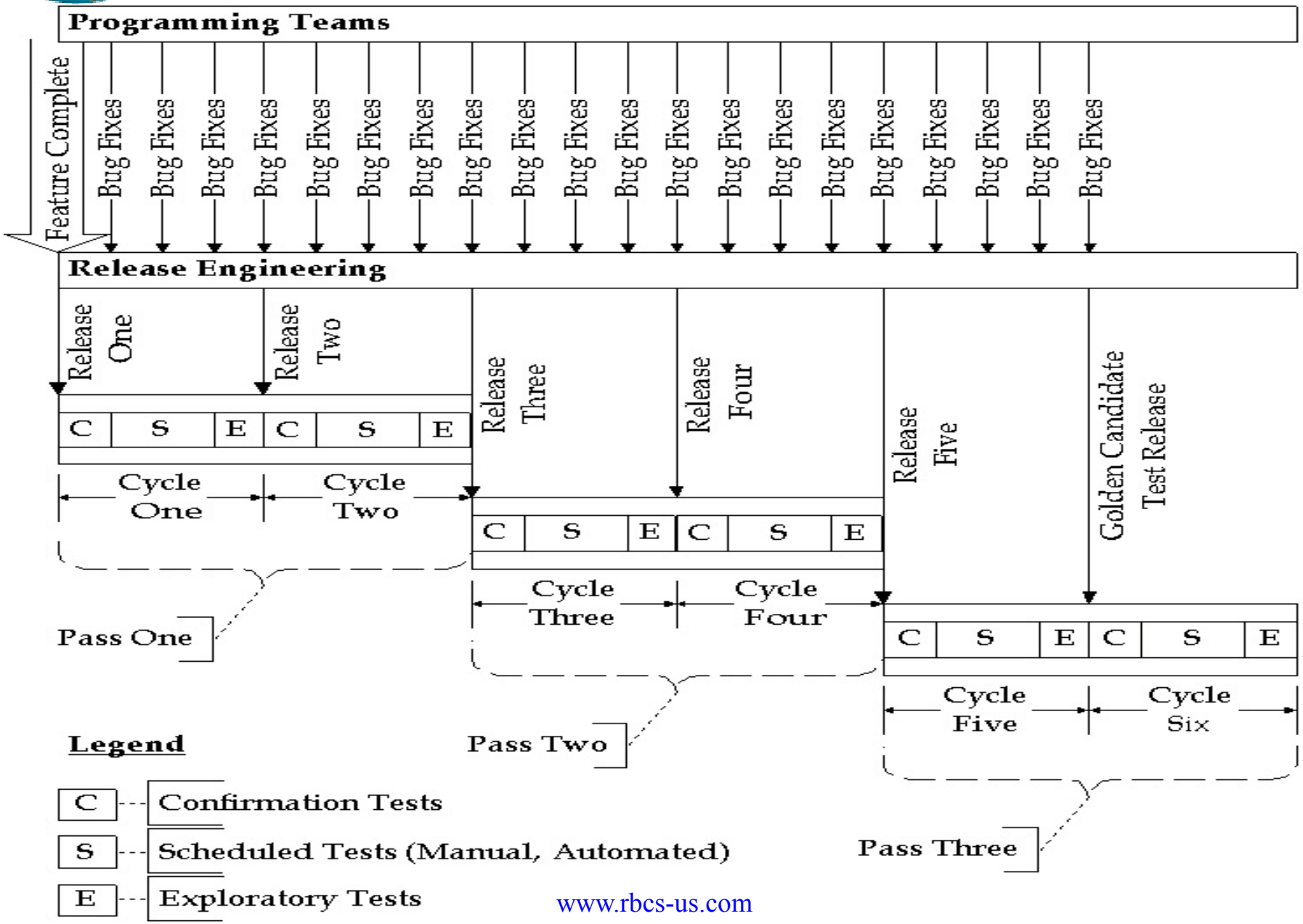
- Assume two people can get the job done in half the time
- Overload tools or the test environment
- Forget setup and support tasks for environments and tools

Predicting Test Execution Time

- When will you be done *executing the tests*?
- Part of the answer is when you'll have run all the planned tests once
  - Total estimated test time (sum for all planned tests)
  - Total person-hours of tester time available per week
  - Time spent testing by each tester
- The other part of the answer is when you'll have found the important bugs and confirmed the fixes
- Estimate total bugs to find, bug find rate, bug fix rate, and closure period (time from find to close) for bugs
  - Historical data really helps
  - Formal defect removal models are even more accurate

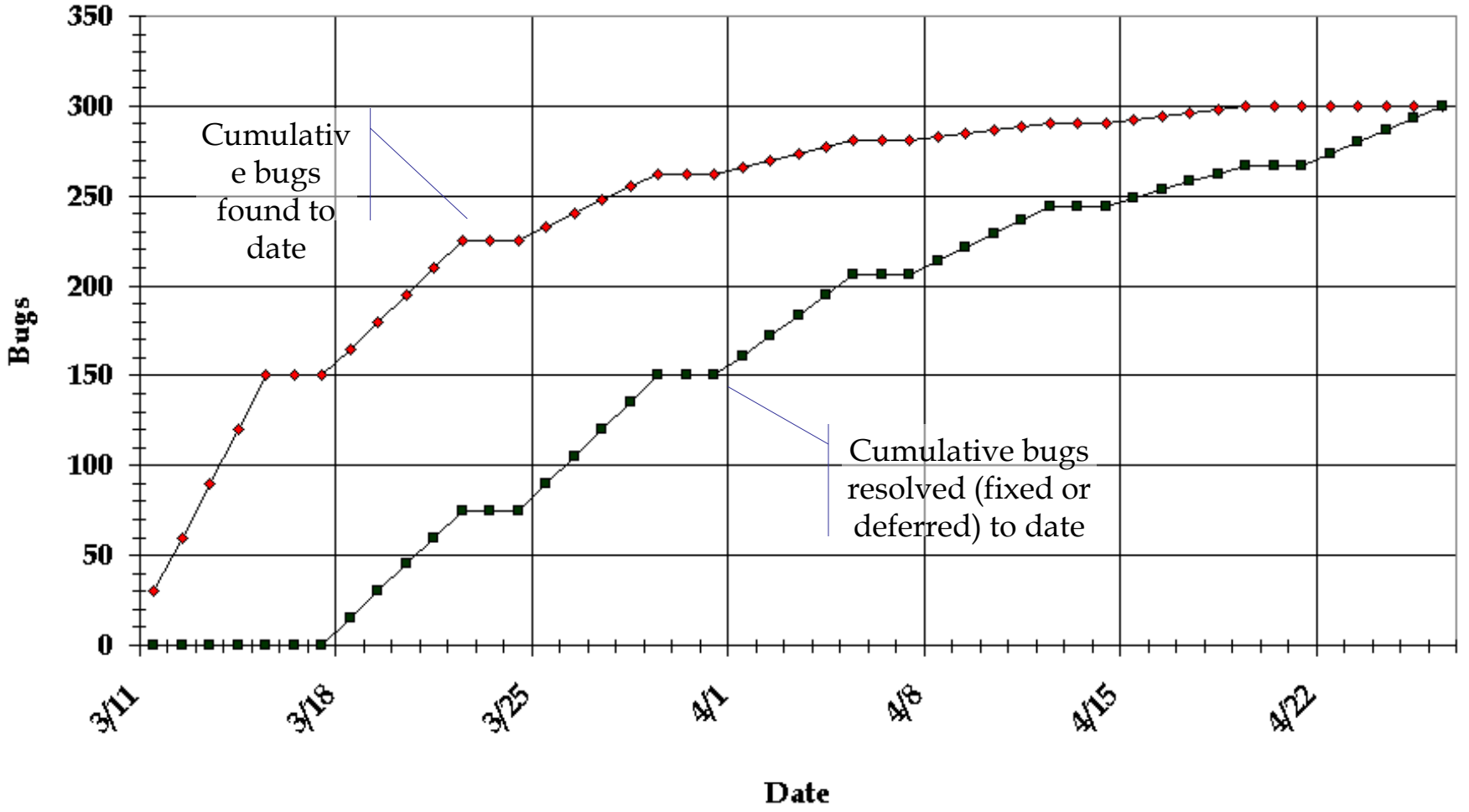
How Long to Run the Tests?

- It depends a lot on how you run tests
  - Scripted vs. exploratory
  - Regression testing strategy (repeat tests or just run once?)
- What I often do
  - Plan for consistent test cycles (tests run per test release) and passes (running each test once)
  - Realize that buggy deliverables and uninstallable builds slow test execution…and plan accordingly
  - Try to understand the amount of confirmation testing, as a large number of bugs leads to lots of confirmation testing
  - Check number of cycles with bug prediction
- Testers spend less than 100% of their time testing
  - E-mail, meetings, reviewing bugs and tests, etc.
  - I plan six hours of testing in a nine-to-ten hour day (contractor)
  - Four hours of testing in an eight hour day is common (employee)



How Long to Get the Bugs Out?

- Using historical data and test cycle model, you can develop a simple model for bugs
- Predicting the total number bugs is hard
  - Subject to many factors and nonlinear effects
  - Common techniques: bugs per developer-day, thousands of source lines of code (KSLOC), or function point (FP)
  - Project size is usually the easiest factor to estimate
- Bug injection rates, fix rates, and closure periods are beyond the test team's control
  - You can predict, but document assumptions
  - The more historical data you have for similar (size, staff, technology, etc.) projects, the more accurate and confident you can be



I created this chart in about one hour using historical data from a couple projects and some simplified models of bug find/fix rates. The more data you have, the more accurate your model can be, and the more you can predict the accuracy.

Conclusions

- Test estimation is one of the most difficult parts of software estimation
- Without good test estimation, insufficient testing will usually occur
- Therefore, test managers need to master proper estimation techniques
- It is possible to properly estimate test effort and duration, by applying known estimation best practices and historical data

To Contact RBCS

For over a dozen years, RBCS has delivered services in consulting, outsourcing and training for software and hardware testing. Employing the industry's most experienced and recognized consultants, RBCS conducts product testing, builds and improves testing groups and hires testing staff for hundreds of clients worldwide. Ranging from Fortune 20 companies to start-ups, RBCS clients save time and money through improved product development, decreased tech support calls, improved corporate reputation and more. To learn more about RBCS, visit www.rbcs-us.com.

Address:	RBCS, Inc. 31520 Beck Road Bulverde, TX 78163-3911 USA
Phone:	+1 (830) 438-4830
Fax:	+1 (830) 438-4831
E-mail:	info@rbcs-us.com
Web:	www.rbcs-us.com