Understanding Test Management
And the Relationship with Quality Management
Test Management and Quality Management

Are test management and quality management the same or different?

How does testing relate to quality?

Confusion abounds, and creates problems for test teams

Let’s look at five key lessons in clearing the confusion and resolving the problems
Key Lessons in Testing and Quality

- Testing is a skilled activity
- Testing has a quantifiable defect detection effectiveness
- Schedule, budget, and quality trade-offs come later in the lifecycle
- Understand bugs in the wild and bugs in captivity
- Integrate testing into quality management
Testing Requires Skills

- Sign of confusion: “Anyone can test. How hard can it be? Just make sure it works before we ship!”
- Amateur testing (i.e., testing done by people other than skilled test professionals) is less effective and efficient.
- Testing involves a mix of technical, application, and test-specific skills.
- Testing should be managed by experienced test managers.
**Case Study: Testing Skills Pay Off**

- One client used former support staff for their testing.
- These people had no testing skill or experience.
- After their testers got ISTQB Foundation certified, one tester found a major efficiency improvement.
- By using equivalence partitioning to eliminate redundant regression tests, over 25% of the test team’s time was freed up.
Case Study: Skills throughout SDLC

Requirements and design reviews (mandatory) lead by independent test team

Project 1
Spec | Dev + UT | CIT | ST

System testing (mandatory) done by development test team

Project 2
Spec | Dev and UT | CIT | ST

Component integration testing (optional) done by programmers and development test team

Project 3
Spec | Dev and UT | ST

Project 4
Spec | Dev and UT | CIT | ST

System integration testing (mandatory) done by independent test team (includes insourced testers) and security team

Project 5
Spec | Dev and UT | ST

Individual programmers develop and unit test (mandatory)

May Release

System Integration Test | UAT

User acceptance testing (mandatory) done by ultimate end-users (usually experts)

July Release

System Integration Test | UAT

Production
Measuring Defect Detection Effectiveness

- Sign of confusion: “We have bugs in Production! The testing failed!”
- Testing has a quantifiable defect detection effectiveness (DDE)
- DDE is the number of defects detected by any test or quality activity, as a percentage of the defects present
- DDE = defects detected/defects present
Defect detection effectiveness (DDE) = defects detected/defects present
Defect removal effectiveness (DRE) = defects removed/defects present
Defects present = defects introduced (this phase) + defects escaped (previous phase)
Defects removed ≤ defects detected
For system test (ST), DDE = ST defects/(ST defects + UAT defects + prod defects)
Case Study: Observed Industry Capability

<table>
<thead>
<tr>
<th>Activity</th>
<th>DDE</th>
<th>Activity</th>
<th>DDE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Requirements Review</td>
<td>65%</td>
<td>Developer Testing (Best)</td>
<td>50%</td>
</tr>
<tr>
<td>Design Review</td>
<td>65%</td>
<td>Professional Testing (Avg)</td>
<td>85%</td>
</tr>
<tr>
<td>Code Review</td>
<td>60%</td>
<td>Professional Testing (Good)</td>
<td>90%</td>
</tr>
<tr>
<td>Developer Testing (Poor)</td>
<td>10%</td>
<td>Professional Testing (Best)</td>
<td>99%</td>
</tr>
<tr>
<td>Developer Testing (Avg)</td>
<td>25%</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Industry figures taken from Capers Jones, *Estimating Software Costs, 3e*, except for the professional testing figures, which derive from RBCS assessments of clients. Professional testing can only achieve high DDE when it’s part of an integrated testing and quality strategy.
Trade-offs in the Lifecycle

Sign of confusion: “We don’t have the time/money to deliver high quality, so we have to settle for good enough.”

In fact, the schedule, budget, and quality trade-offs arise in the later phases of the lifecycle

Because the cost of removing bugs escalates with each phase, early defect removal reduces overall schedule, budget, and quality risk
Case Study: Overwhelmed at the End

- We assessed one client which relied heavily on bug removal in the final phases of testing.
- We found that they had $100M to $250M in avoidable costs of failure.
- Defects that would cost $37 to remove in a requirements review were being removed for $3,700 each in QA testing.
- Since that extra $3,600 represents effort, the trade-off between quality, schedule and budget was created by high escape rates earlier in the lifecycle.
Bugs, Wild and Captive

- Sign of confusion: “Testing is about breaking software.”
- Actually, the software comes to testing with most of the bugs it will have
- The illusion of testing breaking software arises because so many bugs are captured in testing
- Bugs are introduced throughout lifecycle, but primarily in the early activities, while bugs are often removed primarily late in the lifecycle
- A defect removal model, such as the one shown earlier, helps people understand the true nature of bugs, both in the wild and in captivity
Case Study: Typical vs. Best Practices

In typical industry practice, a high number of defects are introduced early in the lifecycle, then removed (at a much higher cost) later in the lifecycle. Best practices focus on early defect removal, with later test phases focusing on building confidence and reducing risk.
Integrate Testing into Quality Management

Sign of confusion: “We miss important areas in our testing, and we don’t have a clear view of the level of quality before we release.”

All testing activities in the lifecycle should have well-defined coverage goals

All test results should consolidate into a single view on the status of the product
## Case Study: Holistic Approach

<table>
<thead>
<tr>
<th>Test Level</th>
<th>Owner</th>
<th>Objectives</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unit</td>
<td>Development</td>
<td>• Early detection of bugs in code units</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Reduce risk of unit failures in production</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Unit testing run before CIT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Unit tests results documented in the feature turnover</td>
</tr>
<tr>
<td>CIT</td>
<td>Development</td>
<td>• Early detection of bugs in unit interfaces</td>
</tr>
<tr>
<td>(Comp Int Test)</td>
<td></td>
<td>• Reduce risk of dataflow and workflow failures in production</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Ensure development ownership for delivering quality features</td>
</tr>
<tr>
<td>CIT</td>
<td>QA</td>
<td>• Early QA validation of completed features</td>
</tr>
<tr>
<td>System/ SIT</td>
<td>QA</td>
<td>• Detect bugs, reduce risk, build confidence in use cases and end-to-end scenarios</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Detect bugs, reduce risk, build confidence in user workflows</td>
</tr>
<tr>
<td>Beta</td>
<td>Customer</td>
<td>• Detect bugs related to customer deployment</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Reduce risk of failing customer business requirements</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Demonstrate readiness for deployment</td>
</tr>
</tbody>
</table>
Conclusion

People confuse test management and quality management, which creates problems.

Key lessons to resolve the confusion and problems:
- Testing is a skilled activity
- Testing has a quantifiable defect detection effectiveness
- Trade-offs come later in the lifecycle
- Bugs in the wild and bugs in captivity
- Integrate testing into quality management

By applying these key lessons to the software engineering process, test and quality professionals can deliver quality software, thoroughly tested.
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