How much should we spend on quality assurance?

Stan Rifkin

Master Systems Inc.
2604B El Camino Real 244
Carlsbad, California 92008 USA
+1 760 729 3388 sr @ Master-Systems.com
Same question as:

- How much quality assurance is enough?
- When should we stop testing?
- What is the relationship between product quality and the quality assurance process?

Answer: It depends.
Going to speak about a new approach: value-based

- Remarks will be brief.
- Reports on the work of Barry Boehm and his PhD student LiGuo Huang, who graduates in a few months.
- Paper will be published in IEEE *Software* this year. Further results will be presented at the International Conference on Software Engineering in May.
- Codification of what some of us already know & do.
- A promising avenue of research, already with some concrete application.
- A way to think.

- The future!
CSC Balanced Scorecard Process

Weighting ensures that appropriate attention is given to your most critical goals and expectations.

Commitments
38.5%

Customer Satisfaction
31.3%

Quality
14%

Productivity
9.9%

Future Value
6.3%

Most critical

Lower priority

Business Goals & Expectations

© Copyright Computer Sciences Corp.
CSC Balanced Scorecard Process

- Brilliant process, based on a clever, seamless synthesis of **many** best practices.
- BUT, what do I do every day to achieve the results?
- What actions should I take in order to achieve the goals?
Enter: Value-based software engineering

- The problems it is trying to solve:
  - Canceled projects - after large investment.
  - Inefficient projects (e.g., Death March)

- Limitations:
  - Method independent.
  - Cannot solve all problems.
  - More notional than detailed today, in general.

- Solution approach
  - Step-by-step directions for selecting important aspects of the product, process, technology, and human resources.
  - Step-by-step guidance on what to do to achieve win-win outcome.
Example: Value of added testing

Reliability/Test Time Tradeoff

Mean Time Between Failures (Hours)

350,000
300,000
250,000
200,000
150,000
100,000
50,000
0

Added Test Time (%)

0 10 20 30 40 50 60

Source: COCOMO II values for RELY, the reliability required of the software product.
What would you do with the additional test time?

### Table 1. Defect Removal Investment Rating Scales

<table>
<thead>
<tr>
<th>Rating</th>
<th>Automated Analysis</th>
<th>Peer Reviews</th>
<th>Execution Testing and Tools</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Very Low</strong></td>
<td>Simple compiler syntax checking.</td>
<td>No peer review.</td>
<td>No testing.</td>
</tr>
<tr>
<td><strong>Low</strong></td>
<td>Basic compiler capabilities</td>
<td>Ad-hoc informal walkthroughs</td>
<td>Ad-hoc testing and debugging.</td>
</tr>
<tr>
<td><strong>Nominal</strong></td>
<td>Compiler extension&lt;br&gt;Basic requirements and design consistency</td>
<td>Well-defined sequence of preparation, review, minimal follow-up.</td>
<td>Basic test, test data management, problem tracking support. Test criteria based on checklists.</td>
</tr>
<tr>
<td><strong>High</strong></td>
<td>Intermediate-level module and inter-module; Simple requirements/design</td>
<td>Formal review roles with well-trained participants and using basic checklists, follow up.</td>
<td>Well-defined test sequence tailored to organization. Basic test coverage tools, test support system. Basic test process management.</td>
</tr>
<tr>
<td><strong>Very High</strong></td>
<td>More elaborate requirements/design&lt;br&gt;Basic distributed-processing and temporal analysis, model checking, symbolic execution.</td>
<td>Basic review checklists, root cause analysis.&lt;br&gt;Formal follow-up using historical data on inspection rate, preparation rate, fault density.</td>
<td>More advanced test tools, test data preparation, basic test oracle support, distributed monitoring and analysis, assertion checking. Metrics-based test process management.</td>
</tr>
</tbody>
</table>

ROI on VBSE testing: There is an optimum, given the goal

Source: Huang & Barry Boehm, op. cit.
Comparing Value-Based Testing vs. Value-Neutral Testing

Source: Huang & Boehm, op. cit.
Risk exposure = Sum over all events of
[Probability of event x size (impact) of event]

Risk Exposure = \( P(L) \times S(L) \)

- high \( P(L) \): inadequate plans
- high \( S(L) \): major problems
  (oversights, delays, rework)
- low \( P(L) \): thorough plans
- low \( S(L) \): minor problems

Source for this slide and the following four: Many of Barry Boehm’s presentations and last year’s SPIN presentation by Stan Rifkin, “What is the best way to develop software? Continuing the conversation about agility and plan-driven methods,” June 2005.
- Loss due to inadequate plans
- Loss due to market share erosion

\[ RE = P(L) \times S(L) \]

- **High** \( P(L) \): inadequate plans
  - Major problems (oversights, delays, rework)
- **High** \( S(L) \): value capture delays
- **Low** \( P(L) \): thorough plans
  - Minor problems
- **Low** \( S(L) \): early value capture

**Example (cont.)**

- Loss due to inadequate plans
- Loss due to market share erosion
Example RE Profile: When to Ship

- Sum of Risk Exposures

\[ RE = P(L) \times S(L) \]

- **Low P(L):** thorough plans
  - Low S(L): minor problems

- **High P(L):** plan breakage, delay
  - High S(L): value capture delays

- **Sweet Spot**
  - High P(L): inadequate plans
    - High S(L): major problems
    - (oversights, delays, rework)

- **When to Ship**
  - Low P(L): few plan delays
    - Low S(L): early value capture
  - Low P(L): thorough plans
    - Low S(L): minor problems
Plan-Driven Home Ground

Time and Effort Invested in Plans

$RE = P(L) \times S(L)$

Higher $S(L)$:
- large system rework

Mainstream Sweet Spot

Plan-Driven Sweet Spot
Time and Effort Invested in Plans

\[ RE = P(L) \times S(L) \]

Mainstream Sweet Spot

Agile Sweet Spot

Lower S(L): easy rework
Another example: Stakeholder synchronization vs. heads-down work

- If I synchronize often with stakeholders it is costly and I avoid rework.
- If I work with my head down I accomplish a lot, don’t have to give “presentations,” and I might be off-track for quite awhile.

- Is there an optimum mix?
### Process Milestones vs. Software Development Activities

<table>
<thead>
<tr>
<th>Process Milestones</th>
<th>Software Development Activities</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initiate project</td>
<td>Acquire system requirements</td>
</tr>
<tr>
<td>SCS define acceptable &amp; desired values for Q-attributes</td>
<td>Requirement elicitation meeting</td>
</tr>
<tr>
<td></td>
<td>Win-Win negotiation</td>
</tr>
<tr>
<td>Risk analysis &amp; architecture/technology evaluation</td>
<td>Internal prototype evaluation</td>
</tr>
<tr>
<td></td>
<td>External prototype evaluation</td>
</tr>
<tr>
<td>Identify conflicting Q-attributes &amp; perform tradeoff analysis</td>
<td></td>
</tr>
<tr>
<td>SCS adjust acceptable values for Q-attributes</td>
<td>Stakeholder renegotiation</td>
</tr>
<tr>
<td>System top-level design and initial Feasibility Rationale Description (FRD)</td>
<td>System top-level design</td>
</tr>
<tr>
<td><strong>LCO Review</strong></td>
<td>Architecture options internal review</td>
</tr>
<tr>
<td></td>
<td>Architecture options external review</td>
</tr>
<tr>
<td>SCS refine acceptable &amp; desired values for Q-attributes</td>
<td>Requirement elicitation meeting</td>
</tr>
<tr>
<td></td>
<td>Win-Win negotiation</td>
</tr>
<tr>
<td>System detailed design and detailed Feasibility Rationale Description (FRD)</td>
<td>System detailed design &amp; FRD</td>
</tr>
<tr>
<td><strong>LCA Review</strong></td>
<td>Selected architecture internal review</td>
</tr>
<tr>
<td></td>
<td>Selected architecture external review</td>
</tr>
<tr>
<td>Core capability implementation</td>
<td>Core capability implementation</td>
</tr>
<tr>
<td>Value-based core capability testing</td>
<td>Internal core capability testing</td>
</tr>
<tr>
<td><strong>CCD</strong></td>
<td>Internal core capability demo</td>
</tr>
<tr>
<td></td>
<td>On-site core capability demo</td>
</tr>
<tr>
<td>Remaining features implementation</td>
<td>Complete system implementation</td>
</tr>
<tr>
<td><strong>IOC Acceptance Review</strong></td>
<td>On-site System Acceptance Review</td>
</tr>
</tbody>
</table>

**Legend:**
- Life Cycle Objective (LCO)
- Life Cycle Architecture (LCA)
- Core Capability Demo (CCD)
- Initial Operational Capability (IOC)

Source: Applying the Value/Petri Process to ERP Software Development in China, LiGuo Huang et al., ICSE 2006.
## ROI on internal vs. external life cycle activities

<table>
<thead>
<tr>
<th>Process Activity Combinations</th>
<th>ROI</th>
</tr>
</thead>
<tbody>
<tr>
<td>LCO(i) \ LCA(i) \ CCD(i) \ IOC(s)</td>
<td>—</td>
</tr>
<tr>
<td>LCO(s) \ LCA(i) \ CCD(i) \ IOC(s)</td>
<td>6.2</td>
</tr>
<tr>
<td>LCO(i) \ LCA(s) \ CCD(i) \ IOC(s)</td>
<td>2.4</td>
</tr>
<tr>
<td>LCO(i) \ LCA(i) \ CCD(s) \ IOC(s)</td>
<td>0.1</td>
</tr>
<tr>
<td>LCO(s) \ LCA(s) \ CCD(i) \ IOC(s)</td>
<td>6.2</td>
</tr>
<tr>
<td>LCO(s) \ LCA(i) \ CCD(s) \ IOC(s)</td>
<td>5.8</td>
</tr>
<tr>
<td>LCO(i) \ LCA(s) \ CCD(s) \ IOC(s)</td>
<td>2.3</td>
</tr>
<tr>
<td>LCO(s) \ LCA(s) \ CCD(s) \ IOC(s)</td>
<td>5.5</td>
</tr>
</tbody>
</table>

Source: Applying the Value/Petri Process to ERP Software Development in China, LiGuo Huang et al., ICSE 2006.
4 + 1 Framework

How do the values vary with other changes in other variables?
Utility Theory

What values are important & to whom? How is success assured?
Decision Theory

How do dependencies affect value realization?
Dependency Theory

What can I control that impacts value?
Control Theory

Theory W: SCS Win-Win

7 Step process of VBSE

Source: LiGuo Huang, private communication.
Utility theory (for money)

Diminishing marginal returns
Other utility curves

Value Loss vs. System Delivery Time:
(a) Marketplace Competition (Internet Services, Wireless Infrastructure);
(b) Fixed-schedule Event Support; (c) Off-line Data Processing

Source: Huang & Boehm, op. cit.
4 + 1 Framework

Utility Theory
How do the values vary with other changes in other variables?

Decision Theory
How do values impact decision choices?

Theory W: SCS Win-Win
What values are important & to whom? How is success assured?

Dependency Theory
How do dependencies affect value realization?

Control Theory
What can I control that impacts value?

Should each peer review be like the next?
Should each test be like the next?
Should each external & milestone review be like the next?