AGILE BY THE NUMBERS
What are the core issues with software development and maintenance?

Improvement measures (Silver bullets?)

Why do software projects succeed or fail?

Agile by the numbers
  ▪ Is Agile a silver bullet

Some problems in paradise
  ▪ Agile issues
Core Issues with Software

• Cost, Schedule, Quality are hard to manage and are often unpredictable
• Frequently do not meet requirements
  Why does this matter?
• Software is pervasive and life as we know it would cease without it
• Money. A huge cost component for business, government, military, communications, and our personal lives
Core Issues with Software

• What is the desired state for software?
  ▪ Predictable
  ▪ Meet requirements
  ▪ Become more efficient over time (productivity improvement)

• New tools and improvement initiatives are best understood in this context
Silver Bullet: A direct and effortless solution to a problem. An action that cuts through complexity and provides an immediate solution to a problem*

Some software improvement initiatives

- Structured programming
- 3gl/4gl languages
- Case tools
- Code generators
- CMMI
- Cloud computing
- GUI’s
- OO Development
- ERP packages
- SOA
- Internet
• Most measures aimed at software improvement have focused on tools, processes, or both.

<table>
<thead>
<tr>
<th>Tools</th>
<th>Process</th>
</tr>
</thead>
<tbody>
<tr>
<td>3/4 GL Languages</td>
<td>Structured Programming</td>
</tr>
<tr>
<td>Case Tools</td>
<td>CMMI</td>
</tr>
<tr>
<td>Code Generators</td>
<td>OO Programming</td>
</tr>
<tr>
<td>GUI's</td>
<td>ERP Packages</td>
</tr>
<tr>
<td>Internet</td>
<td>Internet</td>
</tr>
<tr>
<td></td>
<td>SOA</td>
</tr>
</tbody>
</table>
• “There is no single development, in either technology or management technique, which by itself promises even one order of magnitude improvement within a decade in productivity, in reliability, in simplicity.”

  - Frederick Brooks in “No Silver Bullet – Essence and Accidents of Software Engineering”
Success or Failure
Best and Worst Projects

- Two studies by author
  - 2006 IT projects
  - 2010 Engineering software projects
- Best projects defined as being one standard deviation ($\sigma$) better than average for both time to market (schedule) and effort expended
- Worst projects were one $\sigma$ worse than average for both time to market and effort
- Projects evaluated on 58 criteria for Tools & Methods, Technical Complexity, Personnel, and Re-use
Best Project/Worst Projects

Worst Projects

Schedule vs Size

Best Projects

Worst Projects

Effort vs Size

Best Projects

QSM © The Intelligence behind Successful Software Projects

(#9) 11/15/2012
Differentiators

- Management Efficiency
- Staff Turnover
- Team Skill
- Motivation
- Cohesiveness
- Communication
- Knowledge
- Overall Difficulty
- Customer Interface
- Documentation Rqmts
- Program. Tools Cap.

Average Value of Metrics

- Mgmt Eff.
- Staff Turnover
- Dev Team Skill
- Motivation
- Cohesiveness
- Communication
- Knowledge
- Overall Difficulty
- Customer Interface
- Doc Rqmts
- Program. Tools Cap.

11/15/2012

QSM - The Intelligence behind Successful Software Projects
Things that Don’t Matter

Data Complexity
Integration Complexity
Hardware Stability
System Software Stability
Overall Tools Capability
Project Mgt Tools Capability
Development Standards Experience

Average Value of Metrics

<table>
<thead>
<tr>
<th>Metric</th>
<th>All Systems</th>
<th>Best Projects</th>
<th>Worst Projects</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Complexity</td>
<td>5.7</td>
<td>5.2</td>
<td>4.0</td>
</tr>
<tr>
<td>Integration Complexity</td>
<td>4.0</td>
<td>4.2</td>
<td>4.0</td>
</tr>
<tr>
<td>H/W Stability</td>
<td>9.0</td>
<td>8.7</td>
<td>8.4</td>
</tr>
<tr>
<td>Sys s/w stability</td>
<td>8.4</td>
<td>7.2</td>
<td>6.0</td>
</tr>
<tr>
<td>Overall Tools</td>
<td>6.0</td>
<td>5.8</td>
<td>5.8</td>
</tr>
<tr>
<td>Proj Mgmt Tools Cap.</td>
<td>5.9</td>
<td>5.8</td>
<td>5.8</td>
</tr>
<tr>
<td>Dev Std Exp.</td>
<td>7.0</td>
<td>6.5</td>
<td>6.5</td>
</tr>
</tbody>
</table>
• Results from both the IT and Engineering projects were very similar
• The biggest differentiators between productive and unproductive projects were in the areas of people, communication, and knowledge
• Many project improvement efforts focus on tools and processes
• An interesting tidbit: Project software languages were not correlated with either Best or Worst projects
The Promise of Agile: Agile Manifesto

- Individuals and Interactions over processes and tools
- Working Software over comprehensive documentation
- Customer collaboration over contract negotiation
- Responding to change over following a plan
- Key traits
  - Frequent delivery
  - Business people and developers work together daily
  - Face to face conversations
The Promise of Agile

• It appears that Agile development embraces the People, Knowledge, and Communication traits that were found in highly successful projects.

• Agile is very focused on the social component of software development.

• So, how well do Agile projects compare to traditional development?
64 recently completed Agile projects
12 different companies
87% business, 7% scientific applications, 6% system software
Team size clustered in 5-10 and 20-50 ranges
Median size 42.9k lines of code
Median effort 47 staff months
Median staff 7.5
Median duration 6.1 months
Principally new development and major enhancements
The blue trend lines in this and subsequent graphs are the QSM business average with plus & minus 1 standard deviation. The red line is the Agile dataset average.
Agile Staffing Observations

- The agile projects use slightly more staff than non-agile business projects although the trend is very similar
Agile and non-Agile projects use nearly the same amount of project effort for projects with similar amounts of delivered functionality.
Agile Schedule Length

Agile projects complete much more rapidly

Agile Schedule Duration

Duration (Months) vs Effective SLOC

Comparison of Projects being Assessed to QSM Business Duration (Months) vs Effective SLOC

<table>
<thead>
<tr>
<th></th>
<th>at Min</th>
<th>at 25% Quartile</th>
<th>at Median</th>
<th>at 75% Quartile</th>
<th>at Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>Benchmark Reference Group:</td>
<td>5640</td>
<td>18833</td>
<td>43970</td>
<td>112444</td>
<td>95264</td>
</tr>
<tr>
<td>QSM Business:</td>
<td>3.45</td>
<td>4.90</td>
<td>6.10</td>
<td>8.07</td>
<td>13.94</td>
</tr>
<tr>
<td>Comparison Data Set:</td>
<td>2.54</td>
<td>3.52</td>
<td>4.22</td>
<td>5.61</td>
<td>9.34</td>
</tr>
<tr>
<td>Projects being Assessed:</td>
<td>-0.91</td>
<td>-1.38</td>
<td>-1.78</td>
<td>-2.46</td>
<td>-4.60</td>
</tr>
</tbody>
</table>

Comparison breakpoints based on min, max, median and quartile values for the data set: Projects being Assessed
Agile Schedule Observations

- Agile projects complete much more quickly than non-agile projects while expending about the same amount of effort (Cost)
- Since schedule is frequently an important project driver, this is a significant advantage
Agile Productivity Index (PI)

Productivity indices for Agile projects were significantly higher than the business average.
Agile Quality

Defects Found in Testing

Effective SLOC (thousands)

Comparison of Projects being Assessed vs QSM Business

Errors (SysInt-Del) vs Effective SLOC

<table>
<thead>
<tr>
<th>Errors (SysInt-Del) Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>at Min Effective SLOC</td>
</tr>
<tr>
<td>at 25% Quartile Effective SLOC</td>
</tr>
<tr>
<td>at Median Effective SLOC</td>
</tr>
<tr>
<td>at 75% Quartile Effective SLOC</td>
</tr>
<tr>
<td>at Max Effective SLOC</td>
</tr>
</tbody>
</table>

Comparison headings based on min, max, median and quartile values for the data set. Projects being Assessed

Agile projects produced fewer defects
In Summary

- Agile projects outperform conventional development in Productivity, Quality, and Time to Market
- Staffing levels are higher; but overall effort is slightly lower while achieving significant schedule compression

<table>
<thead>
<tr>
<th>Typical Sized Agile and Business IT Projects</th>
<th>Agile</th>
<th>Business IT</th>
<th>Difference</th>
<th>%Difference</th>
</tr>
</thead>
<tbody>
<tr>
<td>Size in SLOC</td>
<td>42,900</td>
<td>42,900</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Average Staff</td>
<td>9</td>
<td>7.1</td>
<td>1.9</td>
<td>26.8%</td>
</tr>
<tr>
<td>Devel. Duration (Mths)</td>
<td>4.3</td>
<td>6.1</td>
<td>-1.8</td>
<td>-29.5%</td>
</tr>
<tr>
<td>Effort Months</td>
<td>39</td>
<td>43</td>
<td>-4.0</td>
<td>-9.3%</td>
</tr>
<tr>
<td>Defects (testing)</td>
<td>152</td>
<td>245</td>
<td>-93.0</td>
<td>-38.0%</td>
</tr>
<tr>
<td>Productivity Index</td>
<td>19.93</td>
<td>17.92</td>
<td>2.0</td>
<td>11.2%</td>
</tr>
</tbody>
</table>
Some Problems in Paradise

- Large projects require more process formality
  - Change control & Configuration Management
- Regulatory environment may not be compatible with Agile
- Legal requirements & corporate/enterprise requirements
- Minimum marketable features may be very large on big projects
- Budget and schedule constraints are real and legitimate
• Agile is an effective software development strategy
  - Particularly effective at compressing schedule on small to medium size projects
  - Lower defect levels
• Requires investment in training and practice
• Agile is not a panacea for all software development issues
• A good choice; but not for every situation
Contact Information

Donald Beckett,
Principal Consultant
Quantitative Software Mgt.
don_beckett@qsm.com
T: 360-638-0097
C: 703-785-1408