Achieving Excellence
In Software Engineering

Capers Jones, VP and CTO

Web: www.Namcook.com
Email: Capers.Jones3@Gmail.com

October 18, 2012
SIGNIFICANT SOFTWARE INNOVATIONS

- Software defect severity scale (IBM) 1956
- Automated change management tools 1967
- High-level programming languages 1969
- Software process assessments (IBM) 1970
- Structured coding 1971
- Design and code inspections (IBM) 1972
- Automated project management tools 1973
- Automated cost and quality estimation (IBM) 1974
SIGNIFICANT SOFTWARE INNOVATIONS

• Function point metrics (IBM) 1975
• Joint application design (JAD) (IBM) 1976
• Backfiring LOC to function points (IBM) 1978
• Software reusability 1979
• Commercial software estimating tools 1980
• Object-oriented programming 1981
• Complexity analysis tools 1985
• SEI capability maturing model (CMM/CMMI) 1985
## SIGNIFICANT SOFTWARE INNOVATIONS

<table>
<thead>
<tr>
<th>Innovation</th>
<th>Year</th>
</tr>
</thead>
<tbody>
<tr>
<td>Software development workbenches</td>
<td>1986</td>
</tr>
<tr>
<td>Software maintenance workbenches</td>
<td>1988</td>
</tr>
<tr>
<td>Test coverage analysis tools</td>
<td>1990</td>
</tr>
<tr>
<td>Use cases for requirements</td>
<td>1994</td>
</tr>
<tr>
<td>IBM Orthogonal defect classification</td>
<td>1995</td>
</tr>
<tr>
<td>Commercial software benchmarks</td>
<td>1997</td>
</tr>
<tr>
<td>Automated testing tools</td>
<td>1997</td>
</tr>
<tr>
<td>Six-Sigma for Software</td>
<td>2000</td>
</tr>
<tr>
<td>Launch of Wiki-based collaboration</td>
<td>2001</td>
</tr>
</tbody>
</table>
**SIGNIFICANT SOFTWARE INNOVATIONS**

- Automated static analysis tools 2004
- Goal-question metrics 2005
- Virtualization 2006
- Automated function points (legacy software) 2007
- Data mining for lost requirements 2007
- Automated sizing by pattern matching 2008
- Automated, optimized test case design 2008
- MITRE analysis of software defects 2009
- Early sizing and early risk analysis 2010
### APPROXIMATE DEVELOPMENT METHOD CALENDAR

<table>
<thead>
<tr>
<th>Method</th>
<th>Year</th>
</tr>
</thead>
<tbody>
<tr>
<td>Waterfall development</td>
<td>1962</td>
</tr>
<tr>
<td>Structured development</td>
<td>1975</td>
</tr>
<tr>
<td>Object-Oriented development (OO)</td>
<td>1980</td>
</tr>
<tr>
<td>Rapid Application Development (RAD)</td>
<td>1984</td>
</tr>
<tr>
<td>Iterative development</td>
<td>1985</td>
</tr>
<tr>
<td>Rational Unified Process (RUP)</td>
<td>1986</td>
</tr>
<tr>
<td>Agile development</td>
<td>1997</td>
</tr>
<tr>
<td>Team Software Process (TSP)</td>
<td>2000</td>
</tr>
<tr>
<td>Certified reusable components</td>
<td>2015?</td>
</tr>
</tbody>
</table>
### CALENDAR OF SIGNIFICANT SOFTWARE PROBLEMS

<table>
<thead>
<tr>
<th>Event</th>
<th>Year</th>
</tr>
</thead>
<tbody>
<tr>
<td>Requirements &lt; 50% complete</td>
<td>1966</td>
</tr>
<tr>
<td>Requirements change &gt; 2% per month</td>
<td>1975</td>
</tr>
<tr>
<td>Requirements defects resist testing</td>
<td>1976</td>
</tr>
<tr>
<td>Testing &lt; 60% efficient in finding bugs</td>
<td>1978</td>
</tr>
<tr>
<td>Bad fixes &gt; 7% of all defect repairs</td>
<td>1979</td>
</tr>
<tr>
<td>About 5% of modules contain &gt; 50% of defects</td>
<td>1979</td>
</tr>
<tr>
<td>About 35% of large projects are cancelled</td>
<td>1980</td>
</tr>
<tr>
<td>Most estimates are excessively optimistic</td>
<td>1980</td>
</tr>
<tr>
<td>Average defect removal &lt;85% in U.S.</td>
<td>1980</td>
</tr>
</tbody>
</table>
SOFTWARE EXCELLENCE QUESTIONS

• What are the benefits of software excellence?
• How do you judge software excellence?
• How much can development productivity be improved?
• How much can maintenance productivity be improved?
• How much can schedules be improved?
• How much can quality be improved?
• How much will the improvements cost?
• How long will the improvements take?
• What is the ROI of achieving software excellence?
SOFTWARE EXCELLENCE BENEFITS AND RISKS

Benefits:

• Better customer satisfaction
• Better staff morale
• Reduced risk of project failures
• Early risk abatement
• Improved status with top executives

Risks:

• Spending more than necessary to achieve positive results
• Selecting methods that don’t work for all projects
• Expecting excellence in just one year
• Abandoning success after it has been achieved!
  (>40% of companies drop successful methods after 5 years!)
QUANTITATIVE AND QUALITATIVE GOALS

What It Means to be Excellent in Software Engineering

1. Software project cancellation due to major overruns = zero
2. Software cost overruns < 5% compared to formal budgets
3. Software schedule overruns < 3% compared to formal plans
4. Development productivity > 25 function points per staff month
5. Software reuse of design, code, and test cases averages > 75%
6. Development cost < $500 per function point at delivery (burdened)
7. Development schedules average 25% shorter than industry average
8. Software defect potentials average < 2.5 per function point

9. Software defect removal efficiency averages > 96% for all projects

10. Software delivered defects average < 0.1 per function point

11. Software maintenance assignment scopes > 3,500 function points

12. Annual software maintenance < $75 per function point

13. Customer service: Best of any similar corporation

14. User satisfaction: Highest of any similar corporation

15. Staff morale: Highest of any similar corporation

16. Compensation and benefits: Best in your industry
MOVING TO EXCELLENCE IN SOFTWARE ENGINEERING

• Start with an assessment and baseline to find out what is right and wrong with current practices.

• Commission a benchmark study to compare your performance with best practices in your industry.

• Stop doing what is wrong.

• Do more of what is right.

• Set targets: *Best in Class******, Better than Average*****, Better than Today***.

• Develop a three-year technology plan.

• Include: capital equipment, offices, tools, methods, education, culture, languages and return on investment (ROI).
### TECHNICAL REASONS FOR SOFTWARE FAILURES

<table>
<thead>
<tr>
<th>Unsuccessful Projects</th>
<th>Successful Projects</th>
</tr>
</thead>
<tbody>
<tr>
<td>No automated sizing tools</td>
<td>Automated sizing tools</td>
</tr>
<tr>
<td>No automated estimation tools</td>
<td>Automated estimation tools</td>
</tr>
<tr>
<td>No automated planning tools</td>
<td>Automated planning tools</td>
</tr>
<tr>
<td>No progress reporting</td>
<td>Accurate progress reporting</td>
</tr>
<tr>
<td>Inaccurate cost collection</td>
<td>Accurate progress reporting</td>
</tr>
<tr>
<td>No measurement data</td>
<td>Accurate cost collection</td>
</tr>
<tr>
<td>Inaccurate metrics</td>
<td>Substantial measurement data</td>
</tr>
<tr>
<td>No design reviews</td>
<td>Accurate metrics</td>
</tr>
<tr>
<td>No code inspections</td>
<td>Formal design reviews</td>
</tr>
<tr>
<td>No defect tracking</td>
<td>Formal code inspections</td>
</tr>
<tr>
<td>Informal change control</td>
<td>Formal defect tracking</td>
</tr>
<tr>
<td>Unstable requirements (&gt;30%)</td>
<td>Formal change control</td>
</tr>
<tr>
<td>Stable requirements (&lt; 10%)</td>
<td>Stable requirements (&lt; 10%)</td>
</tr>
</tbody>
</table>
SOCIAL REASONS FOR SOFTWARE FAILURES

**Unsuccessful Projects**
- Excessive schedule pressure
- Severe friction with clients
- Poor communications
- Divisive politics
- Naive senior executives
- Management malpractice
- Technical malpractice
- Untrained Generalists

**Successful Projects**
- Realistic schedule expectation
- Cooperation with clients
- Good communications
- Politics held in check
- Experienced senior executives
- Capable management
- Capable technical staff
- Trained Specialists

---

Quality Assurance
Testing
Planning and Estimating
OTHER CORRELATIONS WITH SOFTWARE FAILURES

**Intermittent Failure Factors**

- Geographic separation of team with inadequate communication
- Multiple sub-contractors involved with inadequate communication
- Extraordinary storage or timing constraints
- Projects using “low bid” as sole contract criterion
- Staffing build up > 15% per month
- Staff attrition > 40% of project team
- Abrupt introduction of new technologies
- Projects by companies that are downsizing
- New executives replace proven methods with latest fads
- Trained personnel retire or change jobs
## U.S. SOFTWARE PERFORMANCE LEVELS

<table>
<thead>
<tr>
<th>PROJECT MANAGEMENT</th>
<th>TECHNICAL STAFFS</th>
<th>SOFTWARE USERS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sizing</td>
<td>Fair</td>
<td>Fair</td>
</tr>
<tr>
<td>Estimating</td>
<td>Poor</td>
<td>Design</td>
</tr>
<tr>
<td>Planning</td>
<td>Fair</td>
<td>Coding</td>
</tr>
<tr>
<td>Tracking</td>
<td>Poor</td>
<td>Reviews</td>
</tr>
<tr>
<td>Measuring</td>
<td>Poor</td>
<td>Testing</td>
</tr>
<tr>
<td>Overall</td>
<td>Poor</td>
<td>Good</td>
</tr>
</tbody>
</table>

**Conclusion:** U. S. technical skills are better than U. S. management skills. Project management and quality are frequent problem areas.
SOFTWARE ENGINEERING FACTORS

Worst-case Scenario

Probability of Selected Outcomes

<table>
<thead>
<tr>
<th>Cancel</th>
<th>Delays</th>
<th>On time</th>
<th>Early</th>
</tr>
</thead>
<tbody>
<tr>
<td>15%</td>
<td>75%</td>
<td>10%</td>
<td>0%</td>
</tr>
</tbody>
</table>

1) Ineffective technologies
   Inadequate tool suites
   Inadequate project management
   Less than 5% component reuse
### SOFTWARE ENGINEERING FACTORS (cont.)

<table>
<thead>
<tr>
<th>Single-factor Scenario</th>
<th>Probability of Selected Outcomes</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Cancel</td>
</tr>
<tr>
<td>2) Ineffective technologies</td>
<td>15%</td>
</tr>
<tr>
<td>Inadequate tool suites</td>
<td></td>
</tr>
<tr>
<td><strong>Adequate project management</strong></td>
<td></td>
</tr>
<tr>
<td>Less than 5% component reuse</td>
<td></td>
</tr>
<tr>
<td>3) Ineffective technologies</td>
<td>14%</td>
</tr>
<tr>
<td><strong>Adequate tool suites</strong></td>
<td></td>
</tr>
<tr>
<td>Inadequate project management</td>
<td></td>
</tr>
<tr>
<td>Less than 5% component reuse</td>
<td></td>
</tr>
<tr>
<td>4) <strong>Effective technologies</strong></td>
<td>13%</td>
</tr>
<tr>
<td>Inadequate tool suites</td>
<td></td>
</tr>
<tr>
<td>Inadequate project management</td>
<td></td>
</tr>
<tr>
<td>Less than 5% component reuse</td>
<td></td>
</tr>
<tr>
<td>5) Ineffective technologies</td>
<td>12%</td>
</tr>
<tr>
<td>Inadequate tool suites</td>
<td></td>
</tr>
<tr>
<td>Inadequate project management</td>
<td></td>
</tr>
<tr>
<td><strong>More than 50% component reuse</strong></td>
<td></td>
</tr>
</tbody>
</table>
### SOFTWARE ENGINEERING FACTORS (cont.)

**Two-factor Scenarios**

<table>
<thead>
<tr>
<th>Scenario</th>
<th>Probability of Selected Outcomes</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Cancel</td>
</tr>
<tr>
<td>6) Ineffective technologies</td>
<td>12%</td>
</tr>
<tr>
<td>Adequate tool suites</td>
<td></td>
</tr>
<tr>
<td>Adequate project management</td>
<td></td>
</tr>
<tr>
<td>Less than 5% component reuse</td>
<td></td>
</tr>
<tr>
<td>7) Effective technologies</td>
<td>10%</td>
</tr>
<tr>
<td>Inadequate tool suites</td>
<td></td>
</tr>
<tr>
<td>Adequate project management</td>
<td></td>
</tr>
<tr>
<td>Less than 5% component reuse</td>
<td></td>
</tr>
<tr>
<td>8) Effective technologies</td>
<td>9%</td>
</tr>
<tr>
<td>Adequate tool suites</td>
<td></td>
</tr>
<tr>
<td>Inadequate project management</td>
<td></td>
</tr>
<tr>
<td>Less than 5% component reuse</td>
<td></td>
</tr>
</tbody>
</table>
## SOFTWARE ENGINEERING FACTORS (cont.)

<table>
<thead>
<tr>
<th>Two-factor Scenarios</th>
<th>Probability of Selected Outcomes</th>
</tr>
</thead>
<tbody>
<tr>
<td>9) Ineffective technologies</td>
<td>Cancel</td>
</tr>
<tr>
<td>Adequate tool suites</td>
<td>7%</td>
</tr>
<tr>
<td>Adequate project management</td>
<td></td>
</tr>
<tr>
<td>More than 50% component reuse</td>
<td></td>
</tr>
<tr>
<td>10) Ineffective technologies</td>
<td>Cancel</td>
</tr>
<tr>
<td>Adequate tool suites</td>
<td>7%</td>
</tr>
<tr>
<td>Adequate project management</td>
<td></td>
</tr>
<tr>
<td>More than 50% component reuse</td>
<td></td>
</tr>
<tr>
<td>11) Effective technologies</td>
<td>Cancel</td>
</tr>
<tr>
<td>Adequate tool suites</td>
<td>6%</td>
</tr>
<tr>
<td>Adequate project management</td>
<td></td>
</tr>
<tr>
<td>More than 50% component reuse</td>
<td></td>
</tr>
<tr>
<td>Three-factor Scenarios</td>
<td>Probability of Selected Outcomes</td>
</tr>
<tr>
<td>---------------------------------------------------------------------------------------</td>
<td>----------------------------------</td>
</tr>
<tr>
<td></td>
<td>Cancel</td>
</tr>
<tr>
<td><strong>12) Effective technologies</strong></td>
<td>5%</td>
</tr>
<tr>
<td>Adequate tool suites</td>
<td></td>
</tr>
<tr>
<td>Adequate project management</td>
<td></td>
</tr>
<tr>
<td>Less than 5% component reuse</td>
<td></td>
</tr>
<tr>
<td><strong>13) Effective technologies</strong></td>
<td>4%</td>
</tr>
<tr>
<td>Adequate tool suites</td>
<td></td>
</tr>
<tr>
<td>Inadequate project management</td>
<td></td>
</tr>
<tr>
<td>More than 50% component reuse</td>
<td></td>
</tr>
<tr>
<td><strong>14) Effective technologies</strong></td>
<td>3%</td>
</tr>
<tr>
<td>Inadequate tool suites</td>
<td></td>
</tr>
<tr>
<td>Adequate project management</td>
<td></td>
</tr>
<tr>
<td>More than 50% component reuse</td>
<td></td>
</tr>
<tr>
<td><strong>15) Ineffective technologies</strong></td>
<td>2%</td>
</tr>
<tr>
<td>Adequate tool suites</td>
<td></td>
</tr>
<tr>
<td>Adequate project management</td>
<td></td>
</tr>
<tr>
<td>More than 50% component reuse</td>
<td></td>
</tr>
</tbody>
</table>
### SOFTWARE ENGINEERING FACTORS (cont.)

#### Best-case Scenario

<table>
<thead>
<tr>
<th>Probability of Selected Outcomes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cancel</td>
</tr>
<tr>
<td>--------</td>
</tr>
<tr>
<td>1%</td>
</tr>
</tbody>
</table>

#### 16) Effective technologies
- Adequate tool suites
- Adequate project management
- More than 50% component reuse

* NOTE: Delays caused by external factors; not by poor quality or internal project problems.
PLANNED VERSUS ACTUAL PROJECT SCHEDULES

Function points raised to the 0.3 power is the average schedule plan.

Function points raised to the 0.4 power is the average schedule result.
SOFTWARE PAPERWORK

Application Size in Function Points

Total Volume of Pages Produced (Requirements, Design, Documentation)

Web applets and Agile projects < 250 words per function point in plans, specifications

Less Paperwork Than Expected

Military software > 4000 words per function point in plans, specs, and manuals

More Paperwork Than Expected
RISK OF PROJECT FAILURE

Application Size in Function Points

Probability of Cancellation

5% 10% 15% 20% 25% 30% 35% 40% 45% 50%

5% 10% 15% 20% 25% 30% 35% 40% 45% 50%

SEI CMM 1 > 40% failures
SEI CMM 3 < 15% failures
SEI CMM 5 < 3% failures
for 10,000 function point projects
# RISKS OF FAILURE OR DELAY BY CMM LEVEL

(Complex projects of 10,000 function points in size)

<table>
<thead>
<tr>
<th>SEI CMM LEVEL</th>
<th>Delay &gt; 1 year</th>
<th>Termination</th>
</tr>
</thead>
<tbody>
<tr>
<td>SEI CMM Level 1</td>
<td>35%</td>
<td>40%</td>
</tr>
<tr>
<td>SEI CMM Level 2</td>
<td>30%</td>
<td>30%</td>
</tr>
<tr>
<td>SEI CMM Level 3</td>
<td>20%</td>
<td>12%</td>
</tr>
<tr>
<td>SEI CMM Level 4</td>
<td>12%</td>
<td>04%</td>
</tr>
<tr>
<td>SEI CMM Level 5</td>
<td>08%</td>
<td>02%</td>
</tr>
</tbody>
</table>
SOFTWARE LIFE EXPECTANCY

After 5 years restructuring and complexity analysis are needed

After 10 years compilers may not be available
ANNUAL SOFTWARE ENHANCEMENTS

Expect about 7% per year new and changed features after the first release.
AVERAGE PRODUCTIVITY RATES (NEW PROJECTS)

- **Maximum productivity**: > 100 function points per staff month
- **Minimum productivity**: < 0.5 function points per staff month
- **Average productivity**: is 8 - 12 function points per staff month
**PRODUCTIVITY RATES FOR ENHANCEMENT SOFTWARE PROJECTS**

![Graph showing productivity rates for enhancement software projects.](image)

- **Function Points per Staff Month**
  - Overhead of base application
  - New features for existing applications
  - Major structural changes

*Application Size in Function Points*

Copyright © 2011 by Capers Jones. All Rights Reserved.
PRODUCTIVITY RATES (OVERALL AVERAGE)

Function Points per Staff Month

Application Size in Function Points

Small Enhancements

Mid-sized projects

Massive new projects
SOFTWARE QUALITY IMPROVEMENT

Defect Removal Efficiency

Defects per FP

U.S. Average

Best in Class

Malpractice

SEI CMM 5

SEI CMM 3

SEI CMM 1
SEVEN STAGES OF SOFTWARE EXCELLENCE

Stage 0: Assessment, Baseline, Benchmark analysis
Stage 1: Focus on Project Management
Stage 2: Focus on Development and Maintenance Methods
Stage 3: Focus on New Tools and Approaches
Stage 4: Focus on Infrastructure
Stage 5: Focus on Reusability
Stage 6: Focus on Industry Leadership
Stage 7: Focus on continuous improvement forever!
## TIME REQUIRED TO ADVANCE FROM STAGE TO STAGE

(Duration in Calendar Months)

<table>
<thead>
<tr>
<th>Stage</th>
<th>&lt;10</th>
<th>11-100</th>
<th>101-1000</th>
<th>&gt;1000</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stage 0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>Assessment/Baseline</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Stage 1</td>
<td>3</td>
<td>6</td>
<td>7</td>
<td>8</td>
</tr>
<tr>
<td>Management</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Stage 2</td>
<td>3</td>
<td>6</td>
<td>9</td>
<td>9</td>
</tr>
<tr>
<td>Methods</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Stage 3</td>
<td>4</td>
<td>4</td>
<td>6</td>
<td>9</td>
</tr>
<tr>
<td>Tools</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Stage 4</td>
<td>3</td>
<td>4</td>
<td>6</td>
<td>9</td>
</tr>
<tr>
<td>Infrastructure</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Stage 5</td>
<td>4</td>
<td>6</td>
<td>8</td>
<td>12</td>
</tr>
<tr>
<td>Reusability</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Stage 6</td>
<td>6</td>
<td>8</td>
<td>9</td>
<td>9</td>
</tr>
<tr>
<td>Leadership</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td>24</td>
<td>36</td>
<td>48</td>
<td>60</td>
</tr>
</tbody>
</table>
## The Quality and Productivity Benefits From Completing Each Stage

<table>
<thead>
<tr>
<th>Stage</th>
<th>Defect Reduction</th>
<th>Productivity Increase</th>
<th>Schedule Compression</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stage 0 Assessment</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Stage 1 Management</td>
<td>-10%</td>
<td>0</td>
<td>-10%</td>
</tr>
<tr>
<td>Stage 2 Methods</td>
<td>-50%</td>
<td>25%</td>
<td>-15%</td>
</tr>
<tr>
<td>Stage 3 Tools</td>
<td>-10%</td>
<td>35%</td>
<td>-15%</td>
</tr>
<tr>
<td>Stage 4 Infrastructure</td>
<td>-5%</td>
<td>10%</td>
<td>-5%</td>
</tr>
<tr>
<td>Stage 5 Reusability</td>
<td>-85%</td>
<td>65%</td>
<td>-50%</td>
</tr>
<tr>
<td>Stage 6 Leadership</td>
<td>-5%</td>
<td>5%</td>
<td>-5%</td>
</tr>
<tr>
<td>Overall Results</td>
<td>-90%</td>
<td>350%</td>
<td>-70%</td>
</tr>
</tbody>
</table>
## PROCESS IMPROVEMENT EXPENSES PER CAPITA

<table>
<thead>
<tr>
<th>Stage</th>
<th>Small (≤ 100 staff)</th>
<th>Medium (100-1000)</th>
<th>Large (&gt; 1000 staff)</th>
<th>SEI CMM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stage 0 Assessment</td>
<td>$125</td>
<td>$150</td>
<td>$250</td>
<td>SEI CMM 1</td>
</tr>
<tr>
<td>Stage 1 Management</td>
<td>$1000</td>
<td>$2500</td>
<td>$3000</td>
<td></td>
</tr>
<tr>
<td>Stage 2 Methods</td>
<td>$1500</td>
<td>$2500</td>
<td>$3500</td>
<td>SEI CMM 2</td>
</tr>
<tr>
<td>Stage 3 Tools</td>
<td>$2500</td>
<td>$3500</td>
<td>$5000</td>
<td></td>
</tr>
<tr>
<td>Stage 4 Infrastructure</td>
<td>$1500</td>
<td>$2000</td>
<td>$3000</td>
<td>SEI CMM 3</td>
</tr>
<tr>
<td>Stage 5 Reusability</td>
<td>$2000</td>
<td>$2500</td>
<td>$3500</td>
<td>SEI CMM 4</td>
</tr>
<tr>
<td>Stage 6 Leadership</td>
<td>$1000</td>
<td>$1000</td>
<td>$2000</td>
<td>SEI CMM 5</td>
</tr>
<tr>
<td>Overall Results</td>
<td>$9625</td>
<td>$14150</td>
<td>$20250</td>
<td></td>
</tr>
</tbody>
</table>
RATES OF PROCESS IMPROVEMENT CORRELATED TO INITIAL RANKING

Year 1 | Year 2 | Year 3 | Year 4 | Year 5
---|---|---|---|---
Excellent 1
Good 2
Average 3
Mediocre 4
Poor 5
### BEST CASE RETURN ON INVESTMENT (ROI)

- Assume improvement costs of about $1,000,000
- Value of better quality $5,000,000
- Value of shorter schedules $4,000,000
- Value of higher productivity $3,000,000
- Value of reduced maintenance $2,000,000
- Value of better customer satisfaction $5,000,000

- TOTAL VALUE $20,000,000 *
- RETURN ON INVESTMENT $20 to $1

* Assumes 2 years of improvements and 3 years of results
**UNSUCCESSFUL PROCESS IMPROVEMENT**

- Assume improvement costs of about $1,000,000
- Value of better quality $100,000
- Value of shorter schedules $100,000
- Value of higher productivity $100,000
- Value of reduced maintenance $100,000
- Value of better customer satisfaction $100,000
- TOTAL VALUE $500,000 *
- RETURN ON INVESTMENT $0.5 to $1

* Assumes 2 years of improvements and 3 years of results
STAGE 0: ASSESSMENT, BASELINE, BENCHMARKS

Key Technologies

- SEI Assessment (Levels 1 through 5)
- Six-Sigma Baseline, Benchmark
- Namcook Assessment, Baseline, Benchmark
- ISO 9001 - 9004 Audit
- TickIT assessment
- Putnam Baseline, Benchmark
- Gartner Baseline, Benchmark
- David’s Baseline, Benchmark
- IFPUG Baseline, Benchmark
- ISBSG Benchmarks (commercially available)
STAGE 1: FOCUS ON PROJECT MANAGEMENT

Key Technologies

> Project Sizing
> Project Schedule Planning
> Project Cost Estimating
> Project Quality Estimating
> Functional Metrics
> Project Measurement
> Project Milestone Tracking
> Package Acquisition
> Risk Analysis
> Value Analysis
STAGE 2: FOCUS ON DEVELOPMENT PROCESSES

Key Technologies

- Early sizing and risk assessment
- Reviews and Inspections
- Automated static analysis
- Joint Application Design (JAD)
- Quality Function Deployment (QFD)
- Six-Sigma methodology
- Agile or XP methodologies
- ISO 9001 - 9004 Certification with caution
- SEI maturity levels (CMM and CMMI)
- Rational Unified Process (RUP)
- Team Software Process (TSP)
- Personal Software Process (PSP)
- Geriatric technologies for legacy systems
STAGE 3: FOCUS ON NEW TOOLS & APPROACHES

• Key Technologies -- New Tools
  > Integrated Tool suites
  > Web and Internet Tools

• Key Technologies -- New Approaches
  > Requirements analysis tools
  > CMM/CMMI plus TSP/PSP
  > Use cases
  > Agile Scrum methods
  > Design tools
  > Object-oriented Methods
  > Static analysis; inspections and automated tools
  > Automated testing tools
  > Reverse Engineering and maintenance tools
STAGE 4: FOCUS ON INFRASTRUCTURE

Key Technologies

> Staff Specialization
> Formal Measurement Organization
> Formal Maintenance Organization
> Formal Quality Assurance Organization
> Formal Testing Organization
> Formal Process Improvement Organization
> Improved Hiring Practices
> Improved Compensation Plans
> Competitive Analysis
> Outsource Analysis
STAGE 5:  FOCUS ON REUSABILITY

Key Technologies

- Reusable Architectures
- Reusable Requirements
- Reusable Designs
- Reusable Interfaces
- Reusable Source Code
- Reusable Plans
- Reusable Estimates
- Reusable Data
- Reusable Human Interfaces
- Reusable Test Plans
- Reusable Test Cases
- Reusable Documentation

High quality reuse has best ROI of any technology:
> $40 per $1 expended.

Low quality reuse has worst ROI of any technology:
> - $15 for every $1 expended.
STAGE 6: FOCUS ON INDUSTRY LEADERSHIP

Key Technologies

> Baldrige Award
> Deming Prize
> SEI CMMI Level 5 for major software sites
> Best 100 Companies to Work For
> Market share grows > 20% from baseline
> Time to market better than competitors by > 30%
> Acquisition of Competitors
> Become a Software Outsourcer
STAGE 7: Keeping Excellence After Achieving Excellence

Key Technologies

- Measure results of every project
- Produce monthly reports for managers and teams
- Produce annual reports for top executives
- Publicize results to clients and media
- Train new hires in best practices
- Inform new executives of best practices!!
- Insist on best practices with contractors
- Set targets for annual improvements every year
- Do not abandon success once it achieved!!
ATTRIBUTES OF BEST IN CLASS COMPANIES

1. Good project management
2. Good technical staffs
3. Good support staffs
4. Good measurements
5. Good organization structures
6. Good methodologies
7. Good tool suites
8. Good environments
GOOD PROJECT MANAGEMENT

• Without good project management the rest is unachievable

• Attributes of project good management:
  – Fairness to staff
  – Desire to be excellent
  – Strong customer orientation
  – Strong people orientation
  – Strong technology orientation
  – Understands planning and estimating tools
  – Can defend accurate estimates to clients and executives
  – Can justify investments in tools and processes
GOOD SOFTWARE ENGINEERING TECHNICAL STAFFS

• Without good engineering technical staffs tools are not effective

• Attributes of good technical staffs:
  – Desire to be excellent
  – Good knowledge of applications
  – Good knowledge of development processes
  – Good knowledge of quality and defect removal methods
  – Good knowledge of maintenance methods
  – Good knowledge of programming languages
  – Good knowledge of software engineering tools
  – Like to stay at the leading edge of software engineering
GOOD SUPPORT STAFFS

• Without good support technical staffs and managers are handicapped
• Support staffs > 30% of software personnel in leading companies
• Attributes of good support staffs:
  – Planning and estimating skills
  – Measurement and metric skills
  – Writing/communication skills
  – Quality assurance skills
  – Data base skills
  – Network, internet, and web skills
  – Graphics and web-design skills
  – Testing and integration skills
  – Configuration control and change management skills
GOOD SOFTWARE MEASUREMENTS

• Without good measurements progress is unlikely

• Attributes of good measurements:
  – Function point analysis of entire portfolio
  – Annual function point benchmarks
  – Life-cycle quality measures
  – User satisfaction measures
  – Development and maintenance productivity measures
  – Soft factor assessment measures
  – Hard factor measures of costs, staffing, effort, schedules
  – Measurements used as management tools
GOOD ORGANIZATION STRUCTURES

• Without good organization structures progress is unlikely

• Attributes of good organization structures:
  – Balance of line and staff functions
  – Balance of centralized and decentralized functions
  – Organizations are planned
  – Organizations are dynamic
  – Effective use of specialists for key functions
  – Able to integrate “virtual teams” at remote locations
  – Able to integrate telecommuting
GOOD PROCESSES AND METHODOLOGIES

• Without good processes and methodologies tools are ineffective

• Attributes of good methodologies:
  – Flexible and useful for both new projects and updates
  – Scalable from small projects up to major systems
  – Versatile and able to handle multiple kinds of software
  – Efficient and cost effective
  – Evolutionary and able to handle new kinds of projects
  – Unobtrusive and not viewed as bureaucratic
  – Transferable to new hires, contractors, consultants
GOOD TOOL SUITES

• Without good tool suites, management and staffs are handicapped

• Attributes of good tool suites:
  – Both project management and technical tools
  – Quality tools (static analysis; testing, etc. are critical)
  – Functionally complete
  – Mutually compatible
  – Easy to learn
  – Easy to use
  – Tolerant of user errors
  – Secure
GOOD ENVIRONMENTS AND ERGONOMICS

• Without good office environments productivity is difficult

• Attributes of good environments and ergonomics:
  – Private office space for knowledge workers
    (> 90 square feet; > 6 square meters)
  – Avoid small or crowded cubicles with 3 or more staff
  – Adequate conference and classroom facilities
  – Excellent internet and intranet communications
  – Excellent communication with users and clients
MOST EFFECTIVE PROCESS IMPROVEMENT METHODS

1. Defect removal efficiency measurements
2. Function point productivity and quality measurements
3. Automated static analysis (C, Java, COBOL, SQL etc.)
4. Formal design and code inspections
5. Early sizing and early risk assessments
6. Joint Application Design (JAD) for requirements
7. Automated project management tools
8. Automated cost estimating tools
9. Automated complexity analysis and reduction tools
10. Automated change control tools
11. CMM and CMMI, TSP and PSP
12. Six-Sigma for software
SOFTWARE IMPROVEMENT GUIDELINES

**DO**

- Think long range: 3 to 5 years
- Consider all factors:
  - Management
  - Process
  - Tools
  - Organization
  - Skills and training
  - Programming Languages
  - Environment
- Plan expenses of up to $15,000 per staff member
- Consider your corporate culture

**DON’T**

- Expect immediate results
- Concentrate only on Agile methods or any other “silver bullet”
- Expect major improvements for minor expenses
- Ignore resistance to change
REFERENCES TO PROCESS IMPROVEMENT


Humphrey, Watts; Managing the Software Process; Addison Wesley, 1989.

Jones, Capers; Assessments, Benchmarks, and Best Practices; Addison Wesley, 2000.


Jones, Capers & Bonsignour, Olivier; The Economics of Software Quality; Addison Wesley, 2011.


Wiegers, Karl; Creating a Software Engineering Culture; Dorset House, 1996.

Web sites:

IFPUG.ORG  ITMPI.ORG  ISBSG.ORG  ASQ.ORG  SEI.ORG, IEEE.ORG