Mining PSP Data

Dan Burton and Watts Humphrey
Software Engineering Institute
Carnegie Mellon University
Agenda

Overview – the PSP$^\text{SM}$ and PSP data

Defect-removal analysis

Language-impact analysis

Summary and conclusions
Overview - PSP

The Personal Software Process (PSP) is a process framework to guide developers in
• defining their own processes
• planning and tracking their own work
• managing the quality of the products they produce

PSP principles
• Project success is governed by team performance.
• Team performance depends on member performance.
• Member performance is determined by member practices.
• To consistently improve, developers must use defined and measured processes.
The PSP Course

**Team Software Process**
- Teambuilding
- Risk management
- Project planning and tracking

**PSP0**
- Current process
- Basic measures

**PSP1**
- Size estimating
- Test report

**PSP1.1**
- Task planning
- Schedule planning

**PSP2**
- Code reviews
- Design reviews

**PSP2.1**
- Design templates

**PSP0.1**
- Coding standard
- Process improvement proposal
- Size measurement

- Introduces process discipline and measurement
  - 2 or 3 programs

- Introduces estimating and planning
  - 2 or 3 programs

- Introduces quality management and design
  - 4 programs
Overview – PSP Course

In the PSP course, developers write 8 to 10 programs.

The standard PSP programs are 1 or 2 size counters and 7 or 8 statistical calculations.

Average program size is 107.8 lines of code.

Average development time is 3.98 hours per program.
When using the PSP, developers gather and use data.

Time data
• The time in minutes spent by development task
• Interruption time is not counted.

Size data
• Product size in db elements, pages, LOC, etc.
• Categories: base, added, deleted, modified, reused

Defect data
• All defects removed in compile, test, review, etc.
• Type, phases injected & removed, fix time, description
The PSP course has now been taught for more than ten years.

The SEI has course data on
• 31,140 programs
• 3,355,882 lines of code
• 123,996.53 hours of work
• 221,346 defects

A great deal can be learned from analyzing these data.
PSP Data Analyses

The following charts show summary analyses of
  • defect injection and removal rates
  • productivity
  • development time

Many more analyses can be made.

Based on their PSP data, developers can determine
  • where and how to improve personal practices
  • how and when to best find defects
  • the amount of time required to do a job
  • the likely number of defects to be found in test
# PSP Data by Language

<table>
<thead>
<tr>
<th>Language</th>
<th>Programs</th>
<th>LOC</th>
<th>Hours</th>
<th>Defects</th>
</tr>
</thead>
<tbody>
<tr>
<td>C</td>
<td>4,984</td>
<td>532,529</td>
<td>21,460.8</td>
<td>36,426</td>
</tr>
<tr>
<td>C++</td>
<td>3,255</td>
<td>448,517</td>
<td>14,913.4</td>
<td>30,785</td>
</tr>
<tr>
<td>C#</td>
<td>1,213</td>
<td>163,233</td>
<td>3,696.6</td>
<td>6,661</td>
</tr>
<tr>
<td>VB</td>
<td>1,353</td>
<td>144,621</td>
<td>5,108.5</td>
<td>7,405</td>
</tr>
<tr>
<td>Java</td>
<td>1,383</td>
<td>199,493</td>
<td>6,311.0</td>
<td>11,131</td>
</tr>
<tr>
<td>Ada</td>
<td>286</td>
<td>33,060</td>
<td>1,869.0</td>
<td>3,477</td>
</tr>
<tr>
<td>Total</td>
<td>12,474</td>
<td>1,521,453</td>
<td>53,359.2</td>
<td>95,885</td>
</tr>
</tbody>
</table>
Coding Defects Injected per Hour

![Bar chart showing defects injected per hour in coding for different programming languages (C, C++, C#, VB, Java, Ada) for two programs (Prog. 1 and Prog. 10).]
Defect Fix Time vs. Size

Minutes per Test Defect vs. Program Size

Test Time per Defect (minutes)

Program Size Range (LOC)

© 2006 by Carnegie Mellon University
Unit Test Time vs. Quality

![Bar chart showing the relationship between time per defect and PQI (Proportional Quality Index).](chart.png)
Language Impact Analysis

This analysis looks at the data separated by the following languages and sample sizes.

<table>
<thead>
<tr>
<th>Language</th>
<th>Sample Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>C</td>
<td>455 to 521</td>
</tr>
<tr>
<td>C++</td>
<td>271 to 343</td>
</tr>
<tr>
<td>C#</td>
<td>107 to 127</td>
</tr>
<tr>
<td>VB</td>
<td>119 to 146</td>
</tr>
<tr>
<td>Java</td>
<td>129 to 144</td>
</tr>
</tbody>
</table>
Program Size

Average LOC

LOC

Program Number

C++
C
C#
Java
VB

© 2006 by Carnegie Mellon University
Development Time

Average Development Time

Program Number

Development Minutes

- C++
- C
- C#
- Java
- VB
Total Defects

Average Total Defects

Program Number

© 2006 by Carnegie Mellon University
Productivity

Average Productivity

<table>
<thead>
<tr>
<th>Program Number</th>
<th>Loc/Hours</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>C++</td>
</tr>
<tr>
<td>2</td>
<td>C</td>
</tr>
<tr>
<td>3</td>
<td>C#</td>
</tr>
<tr>
<td>4</td>
<td>Java</td>
</tr>
<tr>
<td>5</td>
<td>VB</td>
</tr>
</tbody>
</table>
Total Defect Density

Average Total Def/KLOC

Program Number

Def/KLOC

C++

C

C#

Java

VB

© 2006 by Carnegie Mellon University
Compile Defect Density

Average Defects/KLOC Removed in Compile

Program Number

Def/KLOC

C++
C
C#
Java
VB

© 2006 by Carnegie Mellon University
Test Defect Density

Average Defects/KLOC Removed in Test

© 2006 by Carnegie Mellon University
Defects/Hour Injected in Design

Average Defects/Hour Injected in Design

Program Number

Defects/Hour

C++
C
C#
Java
VB
Defects/Hour Injected in Code

Average Defects/Hour Injected in Code

© 2006 by Carnegie Mellon University
Defects/Hour Removed in Design Review

Average Defects/Hour Removed in Design Review

- C++
- C
- C#
- Java
- VB

© 2006 by Carnegie Mellon University
Defects/Hour Removed in Code Review

Average Defects/Hour Removed in Code Review

© 2006 by Carnegie Mellon University
Compile + Test Time as % Total

Average Compile + Test Time as % Total Time

© 2006 by Carnegie Mellon University
Yield

Average Yield

Program Number

Yield

C++
C
C#
Java
VB

© 2006 by Carnegie Mellon University
Conclusions

The Personal Software Process (PSP), when properly used provides comprehensive data.

Developers and their teams can use these data to manage their work.

With the aid of PSP data, teams have improved their productivity and shortened development time.

The large volume of PSP data provides a valuable source of information on software engineering practice.
For More Information

Visit the PSP/TSP web site
http://www.sei.cmu.edu/tsp

Contact a TSP transition partner
http://www.sei.cmu.edu/collaborating/partners/trans.part.psp.html

Contact SEI customer relations
Phone, voice mail, and on-demand FAX: 412/268-5800
E-mail: customer-relations@sei.cmu.edu

See the books
Winning with Software, by Watts Humphrey, Addison-Wesley, 2002