Measurement for improving accuracy of estimates: the case study of a small software organisation

Sylvie Trudel
SMEF 2007 – Rome, Italy
May 9th, 2007

Content

- Introduction
- Company
- Process
- Product
- Project estimation
- Improving estimation models
- Preliminary results of the “weighted size” approach
- Conclusion and future work
Introduction

- Accurate software project estimates:
  - Art?
  - Utopia?
  - No, measurement based methodologies!

- Effort and size known to be highly correlated, but…
  - These 2 measures do not guarantee estimation success
  - The team must understand other influencing factors
  - Adding factors to an estimation model may make it less accurate

- Here is the case study of a small Canadian software development company…

Company Overview

- 22 years of existence
- 11 employees
  - All development team members
  - Accounting and house keeping are subcontracted
- 6 active customers
  - 1 large financial organisation ≤ 80% gross revenues
  - 10 years of development of an ERP called “SUM”
- Backlog of projects = 6 to 8 months
Business model

- « Not to exceed » project estimates
  - If actual cost ≥ estimate → invoice=estimate
  - If actual cost < estimate → invoice=actual cost
- Any defect found by the customer is to be fixed at the company’s expense
  - Strong commitment to quality!
- When estimate considered too high by the customer → Project off-shoring to India!
  - Strong motivation for accurate estimates!

Process improvement initiative

- Motivation
  - Missed deadlines on short bi-weekly release cycles
  - Estimates exceeded in 50% of projects
  - Loss potential projects to outsourcing organisations in 2001-2002
- Started PI in 2004, guided by the CMMI
  - Project-oriented
  - 1 project = set of related features
  - 50 hrs < project size < 1300 hrs, average=150 hrs
Process overview

Measurement program

- No measurement plan at first, but…
  - They were measuring effort and schedule
    - To invoice customer every month
- Fall 2006: start measurement plan
  - Exercise to understand information needs
    - Manager
    - Team members
- Classic “Goal-Question/Indicators-Measure” approach
Measurement plan

- Allows the manager and team members to think about their information needs and the quality of measurement
- Simply documented in Excel (only 3 worksheets)

Goals
- Goal description
- Reason

Example of measurement plan

Indicators
- Indicators (questions)
- Formulas
- Goal it relates to
- Unit of measure
- Source of data
- Responsible
- Where stored
- When measured
- Consumer
- Analysis procedure
- Possible actions

Measures
- Measures
- Scope
- Unit of measure
- Precision
- Who measure?
- Data store
- Data collection procedure
- Quality assurance

Product overview

- User interface layer: 30%
- Business logic layer: 50%
- Data access layer (SQL-Server): 20%
- Database layer (SQL-Server): 30%
- SQL scripts SP & UDF

Relative effort per layer to develop new functionalities

VFP, C# Win, or C# Web

C#
Product release cycle

- 1 product release every 2 weeks
  - 1 release = 1..N features from 1..N projects
  - 1 project = 1..N releases
  - Supplemental releases:
    - Only for urgent feature or bug fixing

<table>
<thead>
<tr>
<th>Mon AM</th>
<th>Tue-Wed AM</th>
<th>Thu</th>
<th>Fri AM</th>
<th>Week-end</th>
</tr>
</thead>
<tbody>
<tr>
<td>If any, fix defects 1hr</td>
<td>Package release 10m</td>
<td>If any, fix defects 1hr</td>
<td>Deployment 30m</td>
<td></td>
</tr>
<tr>
<td>Package 10m</td>
<td>Feature testing 1/2d</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Deploy 30m</td>
<td>Test readiness 5m</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Supervised validation testing 1..4 hrs</td>
<td>Re-testing</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Product quality

- In 2006, 35 product releases
  - 17 releases with ZERO defects
  - 18 releases with a total of 28 defects
    - 1.55 defect / release, all fixed within ½ day
- No bug tracking tool
  - Defects are not “managed”, they are “fixed”
Project estimation

- Before 2005:
  - Task-effort estimation only

- From 2005:
  - 2nd method added based on FSM with COSMIC-FFP, and actual effort, to validate 1st estimate
  - Productivity ranges 1.5 to 6 hours/cfsu. Why?
    - CR not systematically measured nor estimated
    - Once performed and isolated, performance variation ranges -6% to +27%

Improving estimation models: a six-steps approach
Step 1: assess reasons for inaccuracy from product and process

- Ratios initially used to adjust estimation model
  - Add new data movement = 100% of effort
  - Delete data movement = 10% of effort
  - Modify existing data movement = 50% of effort

- Problems
  1. Seemed appropriate only if SW in a single layer
  2. With multi-layers architecture, developing new data groups requires more effort to create when developing the first functional process
  3. When modifying existing data groups and data movements, there is a significant difference of effort due to the number of attributes affected, and thus the 50% ratio for maintenance needed to be redefined

- Considering the developer’s viewpoint was abandoned
- Risk of increasing measurement effort

Step 2: evaluate impact of reuse from software architecture layers

- Developing the 2nd functional process
  - All required database components and many business logic components already exist

<table>
<thead>
<tr>
<th>Software layer</th>
<th>New</th>
<th>Reuse</th>
<th>Minor change</th>
<th>Major change</th>
</tr>
</thead>
<tbody>
<tr>
<td>User interface</td>
<td>30%</td>
<td>15%</td>
<td>10%</td>
<td>30%</td>
</tr>
<tr>
<td>Business logic and data (C#)</td>
<td>50%</td>
<td>10%</td>
<td>10%</td>
<td>30%</td>
</tr>
<tr>
<td>Database layer (SQL)</td>
<td>20%</td>
<td>0%</td>
<td>10%</td>
<td>10%</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td>100%</td>
<td>25%</td>
<td>30%</td>
<td>70%</td>
</tr>
</tbody>
</table>
Step 3: apply reusability factors to data movements

- It takes 1 to 2 seconds to identify movement types and reuse impact per data group per functional process
- 1.5 hour to measure an average project

<table>
<thead>
<tr>
<th>Module</th>
<th>Function</th>
<th>Process</th>
<th>Data Group</th>
<th>Movement Types</th>
<th>Reuse Type</th>
<th>R</th>
<th>X</th>
<th>E</th>
<th>W</th>
<th>FFP Total</th>
<th>Reuse Impact</th>
<th>Weighted Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create email/fax</td>
<td>Display main window</td>
<td>Document header</td>
<td>curDocHeader</td>
<td>Déclencheur de l’action</td>
<td>New</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Create email/fax</td>
<td>Display main window</td>
<td>Table dynamique : Read &amp; Exit</td>
<td></td>
<td>Reuse</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>-1,5</td>
<td>0,5</td>
<td></td>
</tr>
<tr>
<td>Create email/fax</td>
<td>Display main window</td>
<td>Table dynamique : Input &amp; Write</td>
<td></td>
<td>New</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>Create email/fax</td>
<td>Display main window</td>
<td>Message(s) simple(s)</td>
<td></td>
<td>New</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

Step 4: establish estimation models per technology

<table>
<thead>
<tr>
<th>Technology</th>
<th>VFP</th>
<th>C# for Windows</th>
<th>C# Web</th>
</tr>
</thead>
<tbody>
<tr>
<td>Estimation model (hours/WSU)</td>
<td>3.22</td>
<td>3.86</td>
<td>5.15</td>
</tr>
</tbody>
</table>

Initial estimation models based on weighted size units (WSU) per technology

Then, 3 C# projects and 2 VFP projects were measured...
### Step 5: adjust effort estimation with risk factors

- 3 risk factors influencing productivity on certain projects:
  - technology: known or unknown
  - complexity: low, medium, high
  - number of other stakeholders involved: none, third party, one or many vendors
- Risk contingency = % total effort
- No risk perceived in majority of projects
  - So as in the sample of projects

### Step 6: validate effort estimation with actual data

<table>
<thead>
<tr>
<th>#</th>
<th>Technology</th>
<th>FFP</th>
<th>WSU</th>
<th>Original est. (hours)</th>
<th>Actual effort (hours)</th>
<th>Over-run %</th>
<th>Hr/FFP</th>
<th>Hr/WSU</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>C# Win</td>
<td>218</td>
<td>159.0</td>
<td>598</td>
<td>567.4</td>
<td>-5%</td>
<td>2.6</td>
<td>3.6</td>
</tr>
<tr>
<td>2</td>
<td>C# Win</td>
<td>74</td>
<td>53.3</td>
<td>131</td>
<td>109.7</td>
<td>-16%</td>
<td>1.5</td>
<td>2.1</td>
</tr>
<tr>
<td>3</td>
<td>C# Win</td>
<td>124</td>
<td>89.5</td>
<td>223</td>
<td>236.9</td>
<td>6%</td>
<td>1.9</td>
<td>2.6</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Average for C# Win:</td>
<td>2.0</td>
<td>2.8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Variance for C# Win:</td>
<td>0.3</td>
<td>0.6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>VFP</td>
<td>47</td>
<td>42.0</td>
<td>102</td>
<td>78.7</td>
<td>-23%</td>
<td>1.7</td>
<td>1.9</td>
</tr>
<tr>
<td>5</td>
<td>VFP</td>
<td>66</td>
<td>55.5</td>
<td>155</td>
<td>138.3</td>
<td>-11%</td>
<td>2.1</td>
<td>2.5</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Average for VFP:</td>
<td>1.9</td>
<td>2.2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Variance for VFP:</td>
<td>0.1</td>
<td>0.2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Preliminary results of the “weighted size” approach

- Insufficient number of data points, but...
  - Average productivity for C# Windows projects went from 4.5 to 2.0 hrs/size unit
  - C# learning curve was not over
  - “Net negative producing programmer” dismissed
  - Software process is applied consistently
- Productivity difference of C# Win and VFP decreased significantly
  - New business opportunities?
- Perceived tendency to overestimate
  - Desired to a certain extent, due to business model

Conclusion and future work

- Inaccurate estimates vs actual effort
  - Often results of lack of discipline to formalize CRs
- Encouraging variance < 16% on C# Win projects
- Continuously monitor actual performance data → readjust estimation models on a periodic basis, but...
  - If precision of “weighted size” < precision of COSMIC size → use COSMIC size
- Experiments on other formulas for weighted size are underway