DOES ANYONE KNOW THE ROOT CAUSE OF OUR PROJECT’S FAILURE?
DOES ANYONE KNOW THE ROOT CAUSE OF OUR PROJECT’S FAILURE?

I’M A DETERMINIST, SO I’D HAVE TO SAY THE PROBLEM GOES BACK TO THE ORIGIN OF THE UNIVERSE.
DOES ANYONE KNOW THE ROOT CAUSE OF OUR PROJECT’S FAILURE?

I’M A DETERMINIST, SO I’D HAVE TO SAY THE PROBLEM GOES BACK TO THE ORIGIN OF THE UNIVERSE.

WHY ARE YOU LIKE THIS? MY CUBICLE DESTROYED MY ILLUSION OF FREE WILL.
Calibration and Historical Data
Chapter 8

Calibration: used to convert counts to estimates

Example:

Counting defects late in the project

• Given 400 open defects
• 250 defects have been corrected – 2 hours per defect
• Estimate Hours needed to close the remaining defects

$$400 \times 2 = 800 \text{ hours}$$
Calibration Data

Three Kinds:

**Industry Data**
Data from other organizations that develop the same kind of software

**Historical Data**
Past data from the organization doing the estimating

**Project Data**
Data generated earlier in the project being estimated
Historical (your own) Data

Estimates are better using Historical Data… better than using any other type of data – or – estimating technique.

Except…

For small projects, individual capabilities are key
… and they may vary from project to project.
For large projects, organizational influences may dominate
Large projects in trouble may not benefit from adding even capable people
So… what organizational influences?
Organizational Influences

• How complex is the software, what is the execution time constraint, what reliability is required, how much documentation is required, how precededent is the (Cocomo II influence factors)

• Can the organization commit to stable requirements, or must the project team deal with volatile requirements throughout the project?

• Is the project manager free to remove a problem team member from the project, or do the organization’s Human Resources policies make it difficult or impossible to remove a problem employee?

• Is the team free to concentrate on the current project, or are team members frequently interrupted with calls to support production releases of previous projects?
More influences…

- Can the organization add team members to the new project as planned, or does it refuse to pull people off other projects?
- Does the organization support the use of effective design, construction, quality assurance, and testing practices?
- Does the organization operate in a regulated environment (for example, under FAA or FDA regulations) in which certain practices are dictated?
- Can the project manager depend on team members staying until the project is complete, or does the organization have high turnover?

Aside: What does high turnover tell you?
Adding “control knobs”

Influence factors
  e.g. Cocomo
  • Many are related to personnel (req’t analysts, programmers, etc.)
  • Cocomo rating method uses industry wide percentile groupings

For programmers:
  90\textsuperscript{th} percentile
  55\textsuperscript{th} percentile
  25\textsuperscript{th} percentile
  15\textsuperscript{th} percentile
… and then there is the *sell*

Upper management attacks the “fat” in the manager’s estimate

Hypothetical Manager / Executive meeting dialog (page 94)

Suggestion (*demands*) of changes to factors...

- Reduced estimated effort by 23%
- Reduced ratings of engineers by 39%

Benefits of historical data… you don’t need to estimate whether programmers are above or below average.

Example

  Company has averaged 300 to 450 LOC per staff month

  Used (assumed) 400 LOC per month
What data to collect?

Start with a small set (may be all you will need):

- Size (LOC or something else you can count)
- Effort (staff months/years)
- Time (months)
- Defects (grouped by severity/priority)

Collect data from more than one project

Numbers you calibrate and use can be off by a factor of 2!
Lines of Code (LOC)

Yes, there are other “things” to count (later chapters)

But what are the definitional issues?

• Do you count all code or only code that’s included in the released software? (e.g. do you count scaffolding code, mock object code, unit test code, and system test code?)
• How do you count code that’s reused from previous versions?
• How do you count open source code or 3rd party library code?
• Do you count blank lines and comments, or only non-blank, non-comment source lines?
• Do you count class interfaces?
• Do you count data declarations?
• How do you count lines that make up one logical line of code but that are broken across multiple lines for the sake of readability?
Subjectivity

- Creeps in
- Estimates can not control for what went wrong in the last project… assuming what went wrong will not happen again.
- … and what went wrong is most often not something the project manager has control over.
- Remember… by using historical data, you assume that the next project will be just like the last project. … although this data may be the best you can get.
Effort?

- Do you count time in hours, days, or some other unit?
- How many hours per day do you count? Standard 8 hours or actual hours applied to the specific project?
- Do you count unpaid overtime?
- Do you count holidays, vacation, and training?
- Do you make allowances for all-company meetings?
- How do you count time that’s divided across multiple projects?
- How do you count time spent supporting previous releases?
- How do you count time spent supporting sales calls, trade shows, and so on?
- How do you count travel time?
- How do you count fuzzy front-end time—the time spent firming up the software concept before the project is fully defined?
Issues related to Calendar Time

• How do you count time that’s divided across multiple projects?
• How do you count time spent supporting previous releases?
• How do you count time spent supporting sales calls, trade shows, etc.
• How do you count travel time?
• How do you count fuzzy front-end time—the time spent firming up the software concept before the project is fully defined?

NOTE. Don’t forget what you are estimating… that is how you tell what data to collect.
You are not fishing.
Finally, defect estimates may vary by a factor of 2 or 3 depending on what’s counted as a defect.

Do you count:

• All change requests as defects, or only those that are classified as defects rather than feature requests?
• Multiple reports of the same defect as a single defect or as multiple defects?
• Defects that are detected by developers, or only those detected by testers?
• Requirements and design defects that are found prior to the beginning of system testing?
• Coding defects that are found prior to the beginning of alpha or beta testing?
• Reported by users after the SW has been released?
Collect data as you go…

• Short-term memory… makes it difficult to reconstruct the past. During or immediately after.

• Collecting data periodically during the project provides for calibration of “time between…”

  For example: Time between different type categories of problem reports.
What to Calibrate

Here are some examples of models you could create:

• Our developers average $X$ lines of code per staff month.
• A 3-person team can deliver $X$ stories per calendar month.
• Our team is averaging $X$ staff hours per use case to create the use case, and $Y$ hours per use case to construct and deliver the use case.
• Our testers create test cases at a rate of $X$ hours per test case.
• In our environment, we average $X$ lines of code per function point in C# and $Y$ lines of code per function point in Python.
• On this project so far, defect correction work has averaged $X$ hours per defect.
Beware of scaling up (or down) Error

May Diseconomies with small variations in Project Size

<table>
<thead>
<tr>
<th>Team Size</th>
<th>Average Stories Delivered per Calendar Month</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>2 3</td>
<td>12</td>
</tr>
<tr>
<td>4 - 5</td>
<td>22</td>
</tr>
<tr>
<td>6 - 7</td>
<td>31</td>
</tr>
<tr>
<td>8</td>
<td>No Data</td>
</tr>
</tbody>
</table>
Using Project Data to Refine your Estimates

• The sooner you can use Project data for calibration and estimating…

the sooner your estimates will become more accurate.

The “Cone” narrows.

• So… Switch from using organizational (historical) data or Industry-average data as soon as possible.
Calibration with Industry Data

Pages 100 - 101

Figure 8-1 (Industry averages)
25\textsuperscript{th} and 75\textsuperscript{th} percentiles
  
  Effort (staff months) and Schedule (months)
  
  Effort: range from 160 to 50 – factor of 3 difference

Figure 8-2 (Organization’s own historical data)
25\textsuperscript{th} and 75\textsuperscript{th} percentiles
  
  Effort (staff months) and Schedule (months)
  
  Effort: range from 95 to 70 – factor of 1.4 difference
Review of Studies

Estimation accuracy:

• When estimation models were NOT calibrated to the organization’s historical data, expert estimates were more accurate than other models

• When Estimation models were calibrated with the organization’s historical data, the models were as good as or better than expert estimates.

Historical data rather than industry wide data is better… not surprising.