Disciplined Software Engineering
Lecture #4

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213

Sponsored by the U.S. Department of Defense

Lecture #4 Overview - Estimating Software Size - 2
Size estimating overview

The PROBE estimating method

Categorizing object data

The regression method

Process additions
Size Estimating Overview

- Obtain historical size data
- Produce conceptual design
- Subdivide the product into parts
- Do the parts resemble parts in the database?
- Select the database parts most like new ones
- Estimate the new part's relative size
- Sum the estimated sizes of the new parts
- Estimate total product size

Product requirement

Repeat until the product parts are the right size

Repeat for all parts

The PROBE Estimating Method

Start

Conceptual Design

Identify Objects

<table>
<thead>
<tr>
<th>Number of Methods</th>
<th>Object Type</th>
<th>Relative Size</th>
<th>Reuse Categories</th>
</tr>
</thead>
</table>

Calculate Added and Modified LOC

Estimate Program Size

Calculate Prediction Interval

Estimate
PROBE Method Description

The following charts describe the PROBE method.

Use form C39 in Appendix C as a reference during the discussion.

The examples are taken from Table 5.8 in Chapter 5.

Conceptual Design

A conceptual design is needed
• to relate the requirements to the product
• to define the product elements that will produce the desired functions
• to estimate the size of what will be built

For understood designs, conceptual designs can be done quickly.

If you do not understand the design, you do not know enough to make an estimate.
Identify the Objects - 1

Where possible, select application entities.

Judge how many methods each object will likely contain.

Determine the type of the object, i.e.: data, calculation, file, control, etc.

Judge the relative size of each object: very small (VS), small (S), medium (M), large (L), very large (VL).

Identify the Objects - 2

From historical object data, determine the size in LOC/method of each object.

Multiply by the number of methods to get the estimated object LOC.

Judge which objects will be added to the reuse library and note as “New Reused.”
Identify the Objects - 3

When objects do not fit an existing type, they are frequently composites.
• Ensure they are sufficiently refined
• Refine those that are not elemental objects

Watch for new object types

Identifying Objects - Example 1

From Table 5.8, 3 new objects are identified, with their numbers of methods, relative size, and LOC.

<table>
<thead>
<tr>
<th>New Objects</th>
<th>Type</th>
<th>Methods</th>
<th>Size</th>
<th>LOC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Matrix</td>
<td>Data</td>
<td>13</td>
<td>M</td>
<td>115</td>
</tr>
<tr>
<td>Linear System</td>
<td>Calc.</td>
<td>8</td>
<td>L</td>
<td>197</td>
</tr>
<tr>
<td>Linked List</td>
<td>Data</td>
<td>3</td>
<td>L</td>
<td>49</td>
</tr>
</tbody>
</table>
Identifying Objects - Example 2

3 reused objects are also shown in Table 5.8.

New objects to be put in the reuse library are identified by an asterisk, such as Linked List.

The unmodified reused objects are

- Linked List 73
- Data Entry 96

Linked List is an existing 73 LOC object with an added method of 49 LOC.

Estimate Program Size - 1

Total program size consists of
- newly developed code (adjusted with the regression parameters)
- reused code from the library
- base code from prior versions, less deletions

Newly developed code consists of
- base additions (BA) - additions to the base
- new objects (NO) - newly developed objects
- modified code (M) - base LOC that are changed
Estimate Program Size - 2

Calculate the new and changed LOC from the newly developed code
- \( BA+NO+M \)
- use regression to get new and changed LOC

\[
New\&\text{Changed} = \beta_0 + \beta_1 \times (BA + NO + M)
\]

\[
y_k = \beta_0 + \beta_1 \times x_k
\]

The regression parameters are calculated from historical data on prior estimated newly developed (object) LOC and actual new and changed LOC.

Estimate Program Size - 3

Code used from the reuse library should be counted and included in the total LOC size estimate.

Base code consists of:
- LOC from the previous version
- subtract deleted code
- subtract modified code (or it would be counted twice)
Completing the Estimate

The completed estimate consists of:
• the estimated new and changed LOC calculated with the regression parameters
• the 70% and 90% upper prediction interval (UPI) and lower prediction interval (LPI) for the new and changed LOC
• the total LOC, considering base, reused, deleted, and modified code
• the projected new reuse LOC to be added to the reuse library

Completed Example - 1

<table>
<thead>
<tr>
<th>Base Program (B)</th>
<th>695 LOC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Deleted (D)</td>
<td>0 LOC</td>
</tr>
<tr>
<td>Modified (M)</td>
<td>5 LOC</td>
</tr>
<tr>
<td>Base Additions (BA)</td>
<td>0 LOC</td>
</tr>
<tr>
<td>New Objects: NO = 115+197+49 =</td>
<td>361 LOC</td>
</tr>
<tr>
<td>Reused Programs</td>
<td>169 LOC</td>
</tr>
</tbody>
</table>
Completed Example - 2

Use the regression parameters to calculate New and Changed LOC (N):

\[ \text{New\&Changed} = \beta_0 + \beta_1 \times (BA + NO + M) \]

Added code: \( BA + NO + M = 366 \) LOC
New and changed: \( N = 62 + 366 \times 1.3 = 538 \) LOC
Total: \( T = 538 + 695 - 5 + 169 = 1397 \) LOC

To Make Size Estimates, You Need Several Items

Data on historical objects, divided into types

Estimating factors for the relative sizes of each object type

Regression parameters for computing new and changed LOC from:
- estimated object LOC
- LOC added to the base
- modified LOC
Historical Data on Objects

Object size is highly variable
• depends on language
• influenced by design style
• helps to normalize by number of methods

Pick basic types
• logic, control
• I/O, files, display
• data, text, calculation
• set-up, error handling

Estimating Factors for Objects

You seek size ranges for each type that will help you judge the sizes of new objects.

To calculate these size ranges
• take the mean
• take the standard deviation
• very small: VS = mean - 2*standard deviations
• small: S = mean - standard deviation
• medium: M = mean
• large: L = mean + standard deviation
• very large: VL = mean + 2*standard deviations
Log-Normal Distribution

These size ranges assume the object data are normally distributed.

If the data are log-normally distributed, take the log of the data before making the size range calculations.

Then, after computing the size ranges, take the antilog to get the factors in LOC.
Estimating Factors - 1

You have the following data on an object type:
- 1 object, 3 methods, 39 total LOC
- 1 object, 5 methods, 127 total LOC
- 1 object, 2 methods, 64 total LOC
- 1 object, 3 methods, 28 total LOC
- 1 object, 1 method, 23 LOC
- 1 object, 2 methods, 44 total LOC

The LOC per method is:
13, 25.4, 32, 9.333, 23, 22
Estimating Factors - 2

The logs of these data are:
• 2.565, 3.235, 3.466, 2.234, 3.135, 3.091
• the average is 2.954
• the standard deviation is 0.421

The log values of the size ranges are then:
• very large - VL: 2.95 + 2*0.42 = 3.79
• large - L: 2.95 + 0.42 = 3.37
• medium - M: 2.95
• small - S: 2.95 - 0.42 = 2.53
• very small - VS: 2.95 - 2*0.42 = 2.11

Estimating Factors - 3

From these log size ranges, the LOC ranges are obtained by taking the antilog
• very large - VL: \( \exp(3.79) = 44.3 \)
• large - L: \( \exp(3.37) = 29.1 \)
• medium - M: \( \exp(2.95) = 19.1 \)
• small - S: \( \exp(2.53) = 12.6 \)
• very small - VS: \( \exp(2.11) = 8.3 \)

Repeat these calculations for every object type
C++ Object Size Ranges

<table>
<thead>
<tr>
<th>Type</th>
<th>LOC per method</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>VS</td>
</tr>
<tr>
<td>Calculation</td>
<td>2.34</td>
</tr>
<tr>
<td>Data</td>
<td>2.60</td>
</tr>
<tr>
<td>I/O</td>
<td>9.01</td>
</tr>
<tr>
<td>Logic</td>
<td>7.55</td>
</tr>
<tr>
<td>Set-up</td>
<td>3.88</td>
</tr>
<tr>
<td>Text</td>
<td>3.75</td>
</tr>
</tbody>
</table>

The Regression Parameters

Using *estimated* object LOC (x) and *actual* new and changed LOC (y):

\[ \beta_1 = \frac{\sum_{i=1}^{n} x_i y_i - nx_{avg} y_{avg}}{\sum_{i=1}^{n} x_i^2 - n(x_{avg})^2} \]

\[ \beta_0 = y_{avg} - \beta_1 x_{avg} \]
The Prediction Interval - 1

The prediction interval provides a likely range around the estimate

• a 90% prediction interval gives the range within which 90% of the estimates will likely fall
• it is not a forecast, only an expectation
• it only applies if the estimate behaves like the historical data

It is calculated from the same data used to calculate the regression factors.

The Prediction Interval - 2

The lower prediction interval (LPI) and upper prediction interval (UPI) are calculated from the size estimate and the range where

• LPI = Estimate - Range
• UPI = Estimate + Range

\[
\text{Range} = t(\alpha / 2, n-2) \sigma \sqrt{1 + \frac{1}{n} + \frac{(x_k - x_{\text{avg}})^2}{\sum_{i=1}^{n}(x_i - x_{\text{avg}})^2}}
\]
The Prediction Interval - 3

The t distribution is for
• the two-sided distribution (alpha/2)
• n-2 degrees of freedom

Sigma is the standard deviation of the regression line from the data.

\[ \sigma = \sqrt{\frac{1}{n-2} \sum_{i=1}^{n} (y_i - \beta_0 - \beta_1 x_i)^2} \]

The t Distribution

The t distribution
• is similar to the normal distribution
• has fatter tails
• is used in estimating statistical parameters from limited data

t distribution tables
• typically give single-sided probability ranges
• we use two-sided values in the prediction interval calculations
The Single-Sided t Distribution

The Double-Sided t Distribution
t Distribution Values

Statistical tables give the probability value $p$ from minus infinity to $x$

For the single-sided value of the tail (the value of interest), take $1-p$

For the double-sided value (with two tails), take

$1 - 2(1 - p) = 2p - 1$

• look under $p = 85\%$ for a 70\% interval
• look under $p = 95\%$ for a 90\% interval

Prediction Interval Example

Calculate the range from historical data

Range = 235 LOC

Upper prediction interval (UPI)

$UPI = N + \text{range} = 538 + 235 = 773 \text{ LOC}$

Lower prediction interval (LPI)

$LPI = N - \text{range} = 538 - 235 = 303 \text{ LOC}$
PSP1 Additions

The PROBE Script - already covered

The test report:
• to report test plans and results
• helpful for later regression testing

Project plan summary
• LOC/hour - plan, actual, to date - to check estimates for reasonableness
• size estimating calculations
• actual size calculations

Size Estimating Calculations

When completing a size estimate, you start with the following data
• new and changed LOC (N): estimate
• modified (M): estimated
• the base LOC (B): measured
• deleted (D): estimated
• the reused LOC (R): measured or estimated

And calculate
• added (A): N-M
• total (T): N+B-M-D+R
Actual Size Calculations

When determining actual program size, you start with the following data
• the total LOC (T): measured
• the base LOC (B): measured
• deleted (D): counted
• the reused LOC (R): measured or counted
• modified (M): counted

And calculate
• added (A): T-B+D-R
• new and changed (N): A+M

Assignment #4

Using PSP1, write program 4A to calculate the linear regression factors for N pairs of data.

Using your data on programs 1A through 3A, make a size and resource estimate and plan.

Use program 4A to calculate the regression factors for programs 1A through 4A.

Follow the program, assignment, and process specifications in Appendices C and D.
Messages to Remember from Lecture 4
1 - The PROBE method is a structured way to make software size estimates.

2 - It uses your personal size data.

3 - It provides a statistically sound range within which the actual program size will most likely fall.