Logistics

papers reaching you early enough?

web site coming soon (email will give URL)

suggested schedule change

<table>
<thead>
<tr>
<th>Date</th>
<th>Event</th>
</tr>
</thead>
<tbody>
<tr>
<td>27 Jan</td>
<td></td>
</tr>
<tr>
<td>10 Feb</td>
<td></td>
</tr>
<tr>
<td>24 Feb</td>
<td></td>
</tr>
<tr>
<td>10 Mar</td>
<td></td>
</tr>
<tr>
<td>24 Mar</td>
<td>SPRING BREAK</td>
</tr>
<tr>
<td>7 Apr</td>
<td></td>
</tr>
<tr>
<td>21 Apr</td>
<td></td>
</tr>
<tr>
<td>5 May</td>
<td>EXAM WEEK</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Date</th>
<th>Event</th>
</tr>
</thead>
<tbody>
<tr>
<td>27 Jan</td>
<td></td>
</tr>
<tr>
<td>10 Feb</td>
<td></td>
</tr>
<tr>
<td>24 Feb</td>
<td></td>
</tr>
<tr>
<td>10 Mar</td>
<td>3 week gap</td>
</tr>
<tr>
<td>24 Mar</td>
<td>31 Mar</td>
</tr>
<tr>
<td>14 Apr</td>
<td></td>
</tr>
<tr>
<td>28 Apr</td>
<td></td>
</tr>
<tr>
<td>12 May</td>
<td></td>
</tr>
</tbody>
</table>
COCOMO – Barry Boehm

based on (fitted to) data from 63 projects

projects fall into one of three “modes”

- organic
- semi-detached
- embedded

yields estimation of effort in MM (152 work hrs)

three estimation levels, using more parameters

- basic: LOC
- intermediate: LOC & cost drivers
- detailed: LOC & cost drivers & life cycle

accuracy claims on Boehm database

- basic: ±200% 60% of the time
- intermediate: ±20% 68% of the time
- detailed: ±20% 70% of the time
COCOMO equations

basic

\[ MM = c \cdot (KDSI)^k \]

look up \( c \) and \( k \) in a table based on project mode
\( c \) and \( k \) can also be calibrated to particular organization

interm.

\[ MM = c \cdot (KDSI)^k \cdot \prod_{i=1}^{15} W(A_i) \]

each attribute \( A_i \) is a rated for a particular project
attributes assess aspects of the...
— product: reliability, database size, complexity
— execution environment: execution time, storage, &c
— personnel: analyst capability, programmer capability, &c
— project: development practices, software tools, scheduling

\( W \) is a table mapping subjective rating to real-valued weight

detailed

\[ MM = c \cdot (KDSI)^k \cdot \prod_{i=1}^{15} W \left( \prod_{p=1}^{4} T_i(A_{i,p}) \right) \]

each attribute \( A_i \) is assessed at life-cycle phase \( p \)
\( T_i \) are tables mapping i th attribute rating to weight
SLIM – Larry Putnam

SLIM is based on the Rayleigh curve

\[ y = at e^{-at^2} \]

The curve has sharp ramp-up, slow decay.

Many engineering phenomena found to obey this curve:
- \( \frac{d(SLOC)}{dt} \)
- \( \frac{d(\text{found defects})}{dt} \)
- \( \frac{d(\text{manpower})}{dt} \)

Using the Rayleigh curve, Putnam derives the so-called “software equation”:

\[
\text{effort} = \frac{B \cdot \text{size}^3}{\text{time}^4 \cdot \text{prod}^3}
\]

- \( \text{prod} \) is a productivity measure.
- \( \text{size} \) is the size of the developed software.
- \( \text{effort} \) is in person-years.
- \( \text{time} \) is in years.
- \( B \) is the “special skills factor,” a constant that varies with size.

This equation helps in estimating the effort required for software development projects.
Key parameters: PI and MBI

productivity index (PI): \( \text{prod}^3 \)

manpower buildup index (MBI): \( \frac{\text{effort}}{(B \cdot \text{time}^3)} \)

how to get values of PI and MBI?

- can be calculated from historical project data
- answer SLIM tool’s 22 questions for an approximation

Diagram:
- Log effort vs. log time
- Software equation line for given size and PI
- Management cost constraint
- Feasibility region
- Min time / max effort solution
- Management time constraint
previous estimates rely on SLOC
  hard to estimate accurately
  especially hard during early phases of life cycle
  requires technical expertise to estimate

instead Albrecht suggests measure of what
function the system computes
  describes inherent cost of development
  neutral with respect to implementation technology (claim)

strong data-processing bias
Counting system functions

- External input
- External inquiries
- Logical internal files
- External interface files
- External output
Calculating function points

calculate unadjusted function points (UFP)
weights are based on value of the function to the customer
“determined by debate and trial”

calculate the technical complexity factor (TCF)
rate 14 characteristics subjectively from 0 to 5

\[
TCF = 0.65 + 0.01 \cdot \sum_{i=1}^{14} C_i
\]

function points \( FP = UFP \cdot TCF \)
Kemerer estimation study

uses magnitude of relative error (MRE) on MM

<table>
<thead>
<tr>
<th>estimation model</th>
<th>% error mean</th>
<th>% error std. dev.</th>
<th>(R^2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SLIM</td>
<td>772</td>
<td>661</td>
<td>0.88</td>
</tr>
<tr>
<td>COCOMO (basic)</td>
<td>610</td>
<td>685</td>
<td>0.68</td>
</tr>
<tr>
<td>Function points</td>
<td>103</td>
<td>112</td>
<td>0.55</td>
</tr>
<tr>
<td>Estimacs *</td>
<td>85</td>
<td>70</td>
<td>0.13</td>
</tr>
</tbody>
</table>

* Estimacs data uses 9 of the 15 projects

questions that he set out to answer

- models work uncalibrated in new environment? NO
- if not, are they accurate when calibrated? YES
- non-SLOC models as accurate as SLOC? YES
- free models as accurate as proprietary? (can’t tell)

models don’t capture productivity factors well

- basic COCOMO out-predicts intermediate and detailed UFP better predictor of SLOC than FP (\(R^2\) of 0.75 vs. 0.66)
- aren’t these factors just “professional judgement” revisited?
Symon’s function points

criticisms of unadjusted function points
   ratings of simple/average/complex are too simple
   weights are ad-hoc and based on value, not effort
   account of internal complexity is “inadequate and confused”
   function points are not “summable”

criticisms of technical complexity factor
   set of factors need to be adjusted over time (and projects?)
   weight scheme is too simple

contribution 1: new counting scheme
   base internal complexity on transaction entities
   base input and output each on data size
   \[ UFP = N_I W_I + N_E W_E + N_O W_O \]
   weights determined by best fit to case study data
   \[ UFP = 0.44 \cdot W_I + 1.67 \cdot W_E + 0.38 \cdot W_O \]

contribution 2: toss in more complexity factors
Reifer’s function points

criticisms

FP born of data processing, de-emphasizes internal processing
how to handle parallelism, concurrency, synchronization?
how to handle real-time modes and heavy math computing?
how to interpret parameters in real-time/scientific venue?
how to compute FP from requirements early in life cycle?

ASSET function point formula

\[ \text{SIZE} = \text{ARCH} \cdot \text{EXPF} \cdot (\text{LANG} \cdot \text{FP}_A + \text{MV}_N)^a \]

where

— SIZE is code size (SLOC)
— ARCH is an architectural factor (table)
— EXPF is a weighted product of 9 size-influencing factors
— LANG is programming language factor (table)
— FP_A is a new counting scheme, one for scientific, one for real-time
— MV_N is the “normalized math volume” (operations count)
— a is the “reuse factor” (not discussed?)

accuracy claim: ±20% from specification

ASSET-R tool automates calculation