2.5 Resources (FP 11-12)

- people:
  - developers
  - management
  - teams
- methods:
  - OOA, OOD, ...
  - SA, ...
- tools:
  - hardware:
    - computers, ...
    - rooms, ...
  - software:
    - compilers, linkers, ...
    - word processing, ...
    - CASE tools, ...

Resources vs. cost

- quality of resource:
  - how does using the resource improve the process (and, ultimately, the product)?
  - productivity
- often the definitive measure of resource quality is the cost:
  - how much does it cost?
  - can we produce more for the same money?

Productivity

= the rate of output per unit of input used especially in measuring capital growth, and in assessing the effective use of labor, materials and equipment (Oxford English Dictionary)

- productivity = output size / effort
- the most commonly used measure: LOC / MM
- is this a process or a resource measure?
  people + tools + methods

LOC / MM as a productivity measure

- size measurement problems revisited:
  - LOC counting technique
  - but:
    - it takes some effort to write (good) comments
    - it takes some effort to reuse (select reusable element, modify)
    - should we count also discarded code?
  - programming language differences

FP / MM as a productivity measure

- FP = functional (not physical) size of output
- useful for more tasks within the lifecycle
- useful for measuring progress
- more difficult to understand
- more difficult to compute (automation)
- depends on project size

Problems with productivity

- questions:
  - are workdays equal?
  - are people equal?
  - are LOC (or FP) equal?
- productivity in other industries:
  - cost of replication?
  - but in SE: cost of design?
- external measure:
  - depends on tools
  - depends on product
  - productivity measures for all phases?
  - LOC/MM can be used for coding & testing
  - is FP/MM good also for specification & design?
Problems with productivity (cont.)

- quality is not included: "very productive"?
- important quality factors:
  - structural information
  - external attributes (usability, reliability, ...)

No single measure is enough!

Problems in measuring people

- motivation problems:
  - tight project schedules: no time for extra work
  - why should we provide these measures?
  - what are the possible consequences of the measures?

The goals of the measurement must be clear and published.

Teams

- team structure:
  - number of members
  - connectivity measures (Fig. 11.2)
  - number of interfaces
  - amount of time spent communicating
  - strong connection to product structure (Table 11.6)

- personnel experience:
  - large effect
  - ordinal measures: e.g.,
    - no experience
    - taken a course
    - one project, use < 20 h
    - several projects, use < 100 h
  - thoroughly experienced
  - quality of experience?

Tools and methods

- measures are often binary: used / not used
- ordinal measures can be developed:
  - general (e.g. COCOMO):
    - very low - low - nominal - high - very high
  - more specific (e.g. CASE tool):
    - not at all - 0-20% of modules - 20-50% of ... - ...
- used mainly with respect to evaluating effects to productivity and cost

Process prediction (FP 12)

- purpose of measuring:
  - to understand the ongoing software process
  - to predict its future

- an example: cost and effort estimation
  - has been examined for a long time
  - still far from being well covered and understood

- what is a good estimate (of cost or effort)?
  - estimate # optimum!
  - estimate # target!

Estimation accuracy

- actual value $A$
- estimate $E$
- relative error $(A-E)/A$
- magnitude of relative error $|A-E|/A$
- mean relative error (for $n$ projects)
  $\frac{1}{n} \sum (A_i - E_i)/A_i$
- prediction quality: percentage of projects where error is within given limits
- estimation quality factor: measure for estimate improvement during process

Estimates can (and should) improve during the project.
Cost estimation

• effort, time, resources
  – usually personnel costs dominate: effort
• preliminary estimates:
  – difficult to obtain
  – crucial for planning
• later in the project:
  – re-estimation
  – resource re-allocation

problems:
  – is the estimation approach reusable?
  – or is the next project entirely different?
  – what is the quality of the result?
  – can we rely on it?
  – can we automate the estimation procedure?

Cost estimation (cont.)

• current approaches:
  – expert opinion
  – analogy = comparison to previous projects
  – decomposition into subtasks / subprojects
  – model of effort
• can be applied
  – bottom-up
  – top-down
  – models of effort/cost:
  – primary factor & set of cost drivers
  – formula for computing overall cost
  – example: COCOMO
  – constraint models:
    – relationship of parameters over time
    – example: Putnam’s SLIM model

Regression-based models

• collection of data from past projects
• relationships among the measures
• find a regression equation connecting the most significant factors
• if there is significant variation:
  – use factor analysis to find causes of variation
  ⇒ new cost factors in the model

Example: linear regression

• simple model:
  – (estimated) size of new code (in LOC)
  ⇒ total development time
  – collection of historical data on previous tasks
• requires at least three data points that show reasonable correlation (no outliers)
• cruder estimates:
  – actual LOC ⇒ actual development hours
  – LOC and development hours ⇒ productivity ⇒ estimated time and range

COCOMO

• three models:
  – basic:
    – used for feasibility study
  – intermediate:
    – based on first plans on environment, people, tools etc.
  – detailed:
    – used when the structure of the software has been defined
• software can be
  – organic – semi-detached – embedded
• primary factor: size S
• effort: \( E = a S^b \times F \)
  – adjustment factor F is composed of cost drivers (=1 for basic)
• project duration:
  \( D = c E^d \)

COCOMO parameters

• effort parameters:
  – product (3)
    – reliability, database size, complexity
  – process (3)
    – practices, tools, schedule
  – resource (5+4)
    – computer time, storage, virtual machine features
    – personnel capability, experience
  – 15 cost drivers:

<table>
<thead>
<tr>
<th>mode</th>
<th>( a )</th>
<th>( b )</th>
<th>( c )</th>
<th>( d )</th>
</tr>
</thead>
<tbody>
<tr>
<td>organic</td>
<td>2.4</td>
<td>1.05</td>
<td>2.5</td>
<td>0.38</td>
</tr>
<tr>
<td>semi-detached</td>
<td>3.6</td>
<td>1.12</td>
<td>2.5</td>
<td>0.35</td>
</tr>
<tr>
<td>embedded</td>
<td>3.6</td>
<td>1.20</td>
<td>2.5</td>
<td>0.32</td>
</tr>
</tbody>
</table>
COCOMO 2.0

- original COCOMO dates from the 1970ies:
  - different tools, processes, environments, ...
- updated version in 1995
  - risk based approach
  - different size metrics depending on stage
  - takes into account reuse, maintenance, change of requirements
- stage 1:
  - prototyping
  - size in object points
- stage 2:
  - evaluate alternatives
  - size in function points
- stage 3:
  - development
  - like original COCOMO

Putnam’s SLIM model

- constraint model based on empirical data
  - developed in 1978 for the US Army
  - collected data on productivity levels
- software equation
- need of manpower is distributed similarly to a set of Rayleigh curves (see Fig. 12.6)
  - separate curve for each phase
  - number of people changes during project lifetime

Putnam’s SLIM model (cont.)

- size $S = C K^{1/3} t_d^{4/3}$
  - technology factor $C$ (14 cost drivers)
  - total project effort $K$ (in years)
  - elapsed time to delivery $t_d$ (in years)
- manpower acceleration $D_0 = K / t_d^3$
  - depends on project type:
    - small if complex interfaces and dependencies
    - large if similar to previous projects

Problems with cost models

- accuracy
  - typically the predicted and actual values differ a lot
- relationship between size and effort
  - sometimes models give contradicting results
- model input
  - use of parameters that are not available early in the project
- too complex models
  - cost drivers
  - technology factors
- size estimation
  - not measurable early
  - inaccurate estimation

How to improve estimation

- well defined and consistent measures
- calibration
  - understand the model
  - adjust model coefficients and cost drivers
- independent estimation group
- reducing subjectivity of input
- two step estimation:
  - preliminary estimates
  - re-estimation using more information
- better size measures
- locally developed models

See Table 12.5

Table 12.5

Putnam’s SLIM model

- constraint model based on empirical data
  - developed in 1978 for the US Army
  - collected data on productivity levels
- software equation
- need of manpower is distributed similarly to a set of Rayleigh curves (see Fig. 12.6)
  - separate curve for each phase
  - number of people changes during project lifetime

Problems with cost models

- accuracy
  - typically the predicted and actual values differ a lot
- relationship between size and effort
  - sometimes models give contradicting results
- model input
  - use of parameters that are not available early in the project
- too complex models
  - cost drivers
  - technology factors
- size estimation
  - not measurable early
  - inaccurate estimation

How to improve estimation

- well defined and consistent measures
- calibration
  - understand the model
  - adjust model coefficients and cost drivers
- independent estimation group
- reducing subjectivity of input
- two step estimation:
  - preliminary estimates
  - re-estimation using more information
- better size measures
- locally developed models

See Table 12.6

Fig. 12.7