Software Cost Estimation

Main Points
- Quantitative models needed for cost estimation
- What factors determine cost/effort?
- How to relate effort to development time?
- Several cost estimation techniques exist
  - All produce estimates, not reliable figures

Cost Estimation Models Creation
- Models are based on experiments
- All models compute effort (in man months) based on some cost drivers
- Cost is directly proportional to effort
- Models are only good for the same type of project they are based on
- Models compute effort in man months. How do we translate to project duration?
Cost Estimation Models

- Quantitative models (e.g. $E = 2.5 \text{KLOC}^{1.05}$)
- Qualitative models (e.g. expert estimation)
- Agile cost estimation

Underlying Cost Drivers

- Writing less code helps
- Reuse helps
- Quality of people is important
- Tools help
- ...

Algorithmic models

- Base formula: $E = a + b\text{KLOC}^c$
- $c$ usually is around 1
- $c > 1$: diseconomy of scale
- $c < 1$: economy of scale
- This nominal cost is multiplied by a function of a number of cost drivers (volatility of requirements, amount of documentation required, CMM level, quality of people, ...)
Arithmetic Models Issues

- Model parameters need to be calibrated for different environments
- Models typically base their estimation on the number of lines of code. How do we determine the number of lines of code in advance?

Walston-Felix

- \( E = 5.2 \text{KLOC}^{0.91} \)
- One of the early algorithmic models (1977)
- Based on 29 productivity factors
- Each factor is described as being one of three values (high, medium, low)
- The model does not account for cross-effect of factors
- Its form influenced many later models

Some Walston-Felix Productivity Factors

<table>
<thead>
<tr>
<th>Variable</th>
<th>Value of variable</th>
<th>average productivity (LOC)</th>
<th>high – low</th>
<th>(PC)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Complexity of user interface</td>
<td>Low</td>
<td>High</td>
<td>Normal</td>
<td>200</td>
</tr>
<tr>
<td>Seniority during requirements specifications</td>
<td>Low</td>
<td>Medium</td>
<td>High</td>
<td>0.9</td>
</tr>
<tr>
<td>Software changed during design</td>
<td>Low</td>
<td>Medium</td>
<td>High</td>
<td>0.9</td>
</tr>
<tr>
<td>Seniority with application area</td>
<td>Low</td>
<td>Medium</td>
<td>High</td>
<td>0.9</td>
</tr>
<tr>
<td>Qualification, experience of personnel</td>
<td>Low</td>
<td>Low</td>
<td>High</td>
<td>200</td>
</tr>
<tr>
<td>Percentage of programmer participation in design</td>
<td>Low</td>
<td>Medium</td>
<td>High</td>
<td>0.9</td>
</tr>
<tr>
<td>Previous experience with operational computer</td>
<td>Low</td>
<td>Medium</td>
<td>High</td>
<td>0.9</td>
</tr>
<tr>
<td>Previous experience with application area</td>
<td>Low</td>
<td>Medium</td>
<td>High</td>
<td>0.9</td>
</tr>
<tr>
<td>Previous experience with software in greater size and complexity</td>
<td>Low</td>
<td>Medium</td>
<td>High</td>
<td>0.9</td>
</tr>
<tr>
<td>Size of source program to document (pages/week)</td>
<td>Low</td>
<td>Medium</td>
<td>High</td>
<td>0.9</td>
</tr>
</tbody>
</table>
**Walston-Felix Productivity Index**

- Productivity Index: \( I = \text{SUM} (W_i X_i) \) for all 29 factors.
- \( W_i = 0.5 \log (P_i) \)

---

**COCOMO (COnstructive COst MOdel)**

- Around since 1981, and very well-documented
- Basic form: \( E = bKLOC^c \), where \( b \) and \( c \) depend on the type of project:
  - Organic: relatively small and well-known
  - Embedded: inflexible environment with many constraints
  - Semidetached: somewhere in between

---

**Typical constant values:**
- Organic: \( b = 2.4 \), \( c = 1.05 \)
- Semidetached: \( b = 3.0 \), \( c = 1.12 \)
- Embedded: \( b = 3.6 \), \( c = 1.2 \)

- More complex form: takes into account 15 multiplicative cost drivers
Putnam Model

- Based on the computation of the area spanned by a Rayleigh curve
- Suitable for very large projects (larger than 15 man years)

Function Point Analysis (FPA)

- Based on the number of data structures used instead of lines of code:
  - I: number of input types
  - O: number of output types
  - E: number of inquiry types
  - L: number of logical internal files
  - F: number of interfaces
- Then, UFP = 4I + 5O + 10E + 4L + 7F

FPA cnt’d

- Somewhat more complex model: constants depend on complexity level (simple, average, complex)
- Cost drivers (application characteristics) next adjust the value of UFP by at most +/- 40%
- Extensive guidelines for counting function points exist
- Suitable for data-centric business applications
COCOMO2

- Successor to COCOMO
- Based on three, increasingly detailed cost models
- Rather than having 3 modes like COCOMO, COCOMO2 has a more elaborate scaling model

COCOMO2 Models

- Application composition model: counting components of large granularity, using object points (objects are: screens, reports, and the like); FPA-like, with 3 levels of complexity for each object.
- Early design model: uses 7 cost drivers (project characteristics, combinations of cost drivers from the post-architecture version) instead of 3 simple complexity levels
- Post-architecture model: an updated version of COCOMO

Use Case Points

- FPA-like model, based on use cases
- Counting depends on the use case:
  - How many steps in success scenario
  - How many classes in the implementation
  - Complexity of the actors involved
- Next, corrections for the technical and environmental complexity
Difficulties with Models

- Data is seldom collected
- People do not collect numbers, so:
  - These models require calibration which requires data
- Cost (or time) is commonly estimated based on political factors

Common Political Factors

- This project costs the same as the last project
- We have 6 months, so it will take 6 months
- Let's outbid the competition by 10%
- The true estimate of time is hard to sell to the boss, let's arbitrarily revise it downwards
- …and other political estimates

Estimation Guidelines

- Do not mix estimation, planning, and bidding
- Combine methods
- Ask for justification
- Select experts with similar prior experience but be wary of differences among projects
- Accept and assess uncertainty
- Provide learning opportunities
- Try to avoid, or postpone, effort estimation
Cone of uncertainty

Project Duration Calculations
- A lot of consensus between models: $T = 2.5E^{1/3}$
- Individual productivity = $777P^{0.5}$ where $P =$ team size
- Compressing this value has a price:
  - Larger team $\Rightarrow$ higher communication overhead
  - New personnel initially slows down team productivity
- Brooks’ law: “Adding manpower to a late project makes it later”

Impossible region
Agile Cost Estimation

- Estimate size of features in *story points*
- Characterize relative sizes: one feature is twice as large as another one, etc.
- Use a few simple relative sizes, e.g., 1, 2, 4, and 8

Translation of story points to real time: *velocity*: number of function points completed in one iteration

- Start: *yesterday’s weather*: productivity is the same as that for the last project
- If the outcome is wrong: adjust the velocity, *not* the story points