Software cost estimation

◆ Predicting the resources required for a software development process
Objectives

◆ To introduce cost and schedule estimation.
◆ To discuss the problems of productivity estimation.
◆ To describe several cost estimation techniques.
◆ To discuss the utility of algorithmic cost modelling and its applicability in the software process.
Topics covered

◆ Productivity
◆ Estimation techniques
◆ Algorithmic cost modelling
◆ Project duration and staffing
Cost estimation objectives

- To establish a budget for a software project.
- To provide a means of controlling project costs.
- To monitor progress against that budget by comparing planned with estimated costs.
- To establish a cost database for future estimation.
- Cost estimation and planning/scheduling are closely related activities.
Software cost components

- Hardware and software costs.
- Travel and training costs.
- Effort costs (the dominant factor in most projects)
  - salaries of engineers involved in the project
  - costs of building, heating, lighting
  - costs of networking and communications
  - costs of shared facilities (e.g., library, staff restaurant)
  - costs of pensions, health insurance, etc.
Costing and pricing

- Estimates are made to discover the cost, to the developer, of producing a software system.
- There is not a simple relationship between the development cost and the price charged to the customer.
Programmer productivity

- A measure of the rate at which individual engineers involved in software development produce software and associated documentation.
- Essentially, we want to measure useful functionality produced per time unit.
Productivity metrics

- Size related measures based on some output from the software process. This may be:
  - lines of delivered source code
  - object code instructions
  - function-related measures based on an estimate of the functionality of the delivered software:
    » Function-points are the best known of this type of measure
Metric problems

- Estimating the size of the measure.
- Estimating the total number of programmer months which have elapsed.
- Estimating contractor productivity (e.g., documentation team) and incorporating this estimate in overall estimate.
Lines of code

- What is a line of code?
- What programs should be counted as part of the system?
- Assumes linear relationship between system size and volume of documentation.
Cross-language comparisons

- The lower level the language, the more productive the programmer.
- The more verbose the programmer, the higher the productivity.
High and low level languages

Low-level language

High-level language
## System development times

<table>
<thead>
<tr>
<th></th>
<th>Analysis</th>
<th>Design</th>
<th>Coding</th>
<th>Testing</th>
<th>Documentation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Assembly code</td>
<td>3 weeks</td>
<td>5 weeks</td>
<td>8 weeks</td>
<td>10 weeks</td>
<td>2 weeks</td>
</tr>
<tr>
<td>High-level language</td>
<td>3 weeks</td>
<td>5 weeks</td>
<td>8 weeks</td>
<td>6 weeks</td>
<td>2 weeks</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Size</th>
<th>Effort</th>
<th>Productivity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Assembly code</td>
<td>5000 lines</td>
<td>28 weeks</td>
<td>714 lines/month</td>
</tr>
<tr>
<td>High-level language</td>
<td>1500 lines</td>
<td>20 weeks</td>
<td>300 lines/month</td>
</tr>
</tbody>
</table>
Function points

- Based on a combination of program characteristics:
  - external inputs and outputs
  - user interactions
  - external interfaces
  - files used by the system

- A weight is associated with each of these.

- The function point count is computed by multiplying each raw count by the weight and summing all values.
Function points

- Function point count modified by complexity of the project.
- FPs can be used to estimate LOC depending on the average number of LOC per FP for a given language.
- FPs are very subjective - depend on the estimator.
- They cannot be counted automatically.
Productivity estimates

- **Real-time embedded systems**
  - 40-160 LOC/P-month

- **Systems programs**
  - 150-400 LOC/P-month

- **Commercial applications**
  - 200-800 LOC/P-month
## Factors affecting productivity

<table>
<thead>
<tr>
<th>Factor</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application domain experience</td>
<td>Knowledge of the application domain is essential for effective software development. Engineers who understand a domain are likely to be the most productive.</td>
</tr>
<tr>
<td>Process quality</td>
<td>The development process used can have a significant effect on productivity. This is covered in Chapter 31.</td>
</tr>
<tr>
<td>Project size</td>
<td>The larger a project, the more time required for team communications. Less time is available for development so individual productivity is reduced.</td>
</tr>
<tr>
<td>Technology support</td>
<td>Good support technology such as CASE tools, supportive configuration management systems, etc. can improve productivity.</td>
</tr>
<tr>
<td>Working environment</td>
<td>As discussed in Chapter 28, a quiet working environment with private work areas contributes to improved productivity.</td>
</tr>
</tbody>
</table>
Quality and productivity

- All metrics based on volume/unit time are flawed because they do not take quality into account.
- Productivity may generally be increased at the cost of quality.
- It is not clear how productivity/quality metrics are related.
Estimation techniques

- Expert judgment
- Estimation by analogy
- Parkinson’s Law
- Pricing to win
- Algorithmic cost modelling
Expert judgment

◆ One or more experts in both software development and the application domain use their experience to predict software costs.

◆ Process iterates until some consensus is reached.

◆ **Advantages:** Relatively cheap estimation method. Can be accurate if experts have direct experience of similar systems.

◆ **Disadvantages:** Very inaccurate if there are no experts!
Estimation by analogy

- The cost of a project is computed by comparing the project to a similar project in the same application domain.

- **Advantages:** Accurate if project data available.

- **Disadvantages:** Impossible if no comparable project has been tackled. Needs systematically maintained cost database.
Parkinson’s Law

◆ The project costs whatever resources are available.
◆ **Advantages:** No overspending.
◆ **Disadvantages:** System is usually unfinished.
Pricing to win

◆ The project costs whatever the customer has to spend on it.
◆ **Advantages:** You get the contract.
◆ **Disadvantages:** The probability that the customer gets the system he wants is small.
Estimation methods

- Each method has strengths and weaknesses.
- Estimation should be based on several methods.
- If these do not return approximately the same result, there is insufficient information available.
- Pricing to win is sometimes the only applicable method.
Algorithmic cost modelling

- Cost is estimated as a mathematical function of product, project and process attributes whose values are estimated by project managers.
- The function is derived from a study of historical costing data.
- Most commonly used product attribute for cost estimation is LOC (code size).
- Most models are basically similar but with different attribute values.
The COCOMO model

- Developed at TRW, a US defense contractor.
- Based on a cost database of more than 60 different projects.
- Exists in three stages:
  - Basic - Gives a “ball-park” estimate based on product attributes.
  - Intermediate - modifies basic estimate using project and process attributes.
  - Advanced - Estimates project phases and parts separately.
Project classes

- **Organic mode:** Small teams, familiar environment, well-understood applications, no difficult non-functional requirements (EASY).

- **Semi-detached mode:** Project team may have experience mixture, system may have more significant non-functional constraints, organization may have less familiarity with application (HARDER).

- **Embedded Hardware/software systems mode:** Tight constraints, unusual for team to have deep application experience (HARD).
Basic COCOMO Formula

- Organic mode: \( PM = 2.4 \times (KDSI)^{1.05} \)
- Semi-detached mode: \( PM = 3 \times (KDSI)^{1.12} \)
- Embedded mode: \( PM = 3.6 \times (KDSI)^{1.2} \)

Note: KDSI is the number of thousands of delivered source instructions.
COCOMO examples

- **Organic mode project, 32KLOC**
  - PM = 2.4 \( (32)^{1.05} \) = 91 person months
  - TDEV = 2.5 \( (91)^{0.38} \) = 14 months
  - N = 91/15 = 6.5 people

- **Embedded mode project, 128KLOC**
  - PM = 3.6 \( (128)^{1.2} \) = 1216 person-months
  - TDEV = 2.5 \( (1216)^{0.32} \) = 24 months
  - N = 1216/24 = 51
COCOMO assumptions

- Implicit productivity estimate
  - Organic mode = 16 LOC/day
  - Embedded mode = 4 LOC/day

- Time required is a function of total effort NOT team size.

- Not clear how to adapt model to personnel availability.
Intermediate COCOMO

- Takes basic COCOMO as starting point.
- Identifies personnel, product, computer and project attributes which affect cost.
- Multiplies basic cost by attribute multipliers which may increase or decrease costs.
Personnel attributes

- Personnel attributes
  - Analyst capability
  - Programmer capability
  - Programming language experience
  - Application experience

- Product attributes
  - Reliability requirement
  - Database size
  - Product complexity
Computer attributes

- Computer attributes
  - Execution time constraints
  - Storage constraints
  - Computer turnaround time

- Project attributes
  - Modern programming practices
  - Software tools
  - Required development schedule
Attribute choice

- These are attributes which were found to be significant in one organization with a limited size of project history database.
- Other attributes may be more significant for other projects.
- Each organization must identify its own attributes and associated multiplier values.
Model tuning

- All numbers in cost model are organization specific. The parameters of the model must be modified to adapt it to local needs.
- A statistically significant database of detailed cost information is necessary.
Example

- Embedded software system on microcomputer hardware.
- Basic COCOMO predicts a 45 person-month effort requirement
- Attributes = RELY (1.15), STOR (1.21), TIME (1.10), TOOL (1.10)
- Intermediate COCOMO predicts
  - $45 \times 1.15 \times 1.21 \times 1.10 \times 1.10 = 76$ person-months.
- Total cost = $76 \times $7000 = $532,000$
Project planning

- Algorithmic cost models provide a basis for project planning as they allow alternative strategies to be compared.
- Alternative 1: Use more powerful hardware to reduce TIME and STOR attribute multipliers.
- Alternative 2: Invest in support environment.
Management options

A. Use existing hardware, development system and development team

B. Processor and memory upgrade
   - Hardware cost increase
   - Experience decrease

C. Memory upgrade only
   - Hardware cost increase

D. More experienced staff

E. New development system
   - Hardware cost increase
   - Experience decrease

F. Staff with hardware experience
Hardware investment

- Processor capacity and store doubled
  - TIME and STOR multipliers = 1
- Extra investment of $30,000 required
- Fewer tools available
  - TOOL = 1.15
- Total cost = 45*1.24*1.15 * $7000 = $ 449,190
- Cost saving = $83,000
Environment investment

- In addition to hardware costs.
- Reduces turnaround, tool multipliers. Increases experience multiplier.
- $C = 45 \times 0.91 \times 0.87 \times 1.1 \times 1.15 \times 7000 = 315,472$
- Saving from investment = $133,718
Development time estimates

- Organic: $TDEV = 2.5 \ (PM)^{0.38}$
- Semi-detached: $TDEV = 2.5 \ (PM)^{0.35}$
- Embedded mode: $TDEV = 2.5 \ (PM)^{0.32}$
- Personnel requirement: $N = PM/TDEV$
Staffing requirements

- Staff required can’t be computed by diving the development time by the required schedule.
- The number of people working on a project varies depending on the phase of the project.
- The more people who work on the project, the more total effort is usually required.
- Very rapid build-up of people often correlates with schedule slippage.
Rayleigh manpower curves
Key points

◆ Estimate the project cost to the supplier then decide on the price to the customer.

◆ Factors affecting productivity include individual aptitude, domain experience, the development project, the project size, tool support and the working environment.

◆ Prepare cost estimates using different techniques. Estimates should be comparable.
Key points

◆ Algorithmic cost estimation is difficult because of the need to estimate attributes of the finished product.
◆ Algorithmic cost models are a useful aid to project managers as a means of comparing different development options.
◆ The time required to complete a project is not simply proportional to the number of people working on the project.