Software cost estimation

Software development processes are split into a number of separate activities. Cost estimation of software development project focuses on how associating estimates of effort and time with the project activities. Estimation involves answering the following questions:

1. How much effort is required to complete each activity?
2. How much calendar time is needed to complete each activity?
3. What is the total cost of each activity?

Project cost estimation and project scheduling are usually carried out together. The costs of development are primarily the costs of the effort involved, so the effort computation is used in both the cost and the schedule estimate. The initial cost estimates may be used to establish a budget for the project and to set a price for the software for a customer. The total cost of a software development project is the sum of following costs:

1. Hardware and software costs including maintenance.
2. Travel and training costs.
3. Effort costs of paying software developers.

For most projects, the dominant cost is the effort cost. Effort costs are not just the salaries of the software engineers who are involved in the project. The following overhead costs are all part of the total effort cost:

1. Costs of heating and lighting offices.
2. Costs of support staff (accountants, administrators, system managers, cleaners, technicians etc.).
3. Costs of networking and communications.
4. Costs of central facilities (library, recreational facilities, etc.).
5. Costs of social security and employee benefits such as pensions and health insurance.

The aim of software costing is to accurately predict the cost of developing the software. The price of software is normally the sum of development cost and profit. During the development project managers should regularly update their cost and schedule estimates. This helps with the planning process and the effective use of resources. If actual expenditure is significantly greater than the estimates, then the project manager must take some action. This may involve applying for additional resources for the project or modifying the work to be done.

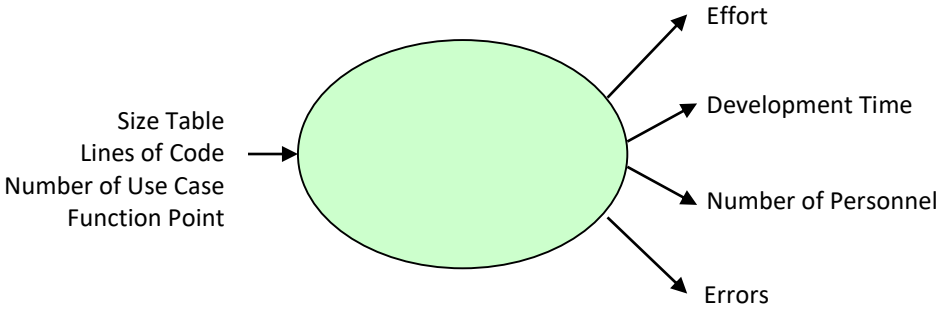
Motivation: The software cost estimation provides:

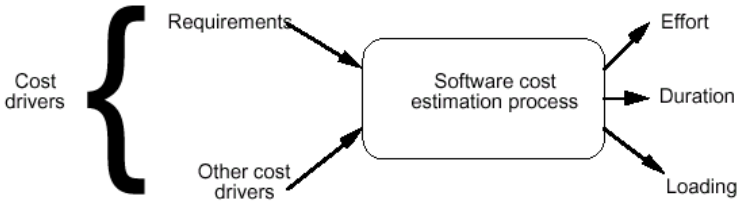
- The vital link between the general concepts and techniques of economic analysis and the particular world of software engineering.
- Software cost estimation techniques also provides an essential part of the foundation for good software management.

Cost of a project

- The cost in a project is due to:
  - the requirements for software, hardware and human resources
  - the cost of software development is due to the human resources needed
  - most cost estimates are measured in *person-months (PM)*
- the cost of the project depends on the nature and characteristics of the project,
- at any point, the accuracy of the estimate will depend on the amount of reliable information we have about the final product.

Software Cost Estimation Process





Software productivity estimation

Productivity estimates help to define the project cost and schedule. In a software development project managers may be faced with the problem of estimating the productivity of software engineers. For any software problem, there may be many different solutions, each of which has different attributes. One solution may execute more efficiently while another may be more readable and easier to maintain and comparing their production rates is very difficult.

Productivity estimates are usually based on measuring attributes of the software and dividing this by the total effort required for development. Software metrics have two main types:

1. Size-related software metrics. These metrics are related to the size of software. The most frequently used size-related metric is lines of delivered source code.
2. Function-related software metrics. These are related to the overall functionality of the software. For example, function points and object points are metrics of this type.

The productivity estimation defined as lines of source code per programmer month is widely used software productivity metric. It is computed by counting the total number of lines of source code divided by the total development time in programmer-months required to complete the project.

In other cases, the total number of function points in a program measures or estimates the following program features:

1. external inputs and outputs,
2. user interactions,
3. external interfaces,
4. files used by the system.

COstructive COst Model (COCOMO)

- The **COstructive COst Model** (COCOMO) is the most widely used software estimation model.
- The COCOMO model predicts the **effort** and **duration** of a project based on inputs relating to the size of the resulting systems and a number of "**cost drives**" that affect productivity.
- COCOMO is defined in terms of three different models:
    i. the **Basic model**,
    ii. the **Intermediate model**, and
    iii. the **Detailed model**.
- The more complex models account for more factors that influence software projects, and make more accurate estimates.

The Development mode: The most important factors contributing to a project's duration and cost is the Development Mode

i. **Organic Mode:** The project is developed in a familiar, stable environment, and the product is similar to previously developed products. The product is relatively small, and requires little innovation.
ii. **Semidetached Mode:** The project's characteristics are intermediate between Organic and Embedded.
iii. **Embedded Mode:** The project is characterized by tight, inflexible constraints and interface requirements. An embedded mode project will require a great deal of innovation.

Features	Organic	Semidetached	Embedded
Organizational understanding of product and objectives	Thorough	Considerable	General
Experience in working with related software systems	Extensive	Considerable	Moderate
Need for softaware conformance with pre-established requirements	Basic	Considerable	Full
Need for softaware conformance with external interface specifications	Basic	Considerable	Full
Concurrent development of associated new hardware and operational procedures	Some	Moderate	Extensive
Need for innovative data processing architecture and algorithms	Low	Medium	High
Product size range (thousands of "delivered source instructions" - KDSI)	< 50 KDSI	< 300 KDSI	ALL

COCOMO in a Coconut-shell

$$E = a(KLOC)^b$$

- Where
  - E is the Effort in staff months
  - a and b are coefficients to be determined
  - KLOC is thousands of lines of code

The Constants

Mode	a	b
Organic	2.4	1.05
Semi-detached	3.0	1.12
Embedded	3.6	1.20

The Modes

- Organic
  - 2-50 KLOC, small, stable, little innovation
- Semi-detached
  - 50-300 KLOC, medium-sized, average abilities, medium time-constraints
- Embedded
  - > 300 KLOC, large project team, complex, innovative, severe constraints

Examples

- Suppose size is 200 KLOC,
  - Organic
    - $2.4(200)^{1.05}$ = 626 staff-months
  - Semi-Detached
    - $3.0(200)^{1.12}$ = 1,133 staff-months
  - Embedded
    - $3.6(200)^{1.20}$ = 2,077 staff-months

Project Duration

$$TDEV = c(E)^d$$

- Where
  - TDEV is time for development
  - c and d are constants to be determined
  - E is the effort

Constants for TDEV

Mode	c	d
Organic	2.5	0.38
Semi-detached	2.5	0.35
Embedded	2.5	0.32

Example

- Picking up from the last example,
  - Organic
    - E = 626 staff months
    - TDEV = $2.5(626)^{0.38}$ = 29 months
  - Semi-detached
    - E = 1,133
    - TDEV = $2.5(1133)^{0.35}$ = 29 months
  - Embedded
    - E = 2077
    - TDEV = $2.5(2077)^{0.32}$ = 29 months

Average Staff Size

$$SS = \frac{E}{TDEV} = \frac{[\text{staff} \cdot \cancel{\text{months}}]}{\cancel{[\text{months}]}} = [\text{staff}]$$

Productivity

$$P = \frac{Size}{E} = \frac{[\text{KLOC}]}{[\text{staff - months}]} = \text{KLOC}/\text{staff - month}$$

Complete Example, Organic

- Suppose an organic project has 7.5 KLOC,
  - Effort $2.4(7.5)^{1.05}$ = 20 staff–months
  - Development time $2.5(20)^{0.38}$ = 8 months
  - Average staff 20 / 8 = 2.5 staff
  - Productivity 7,500 LOC / 20 staff-months = 375 LOC / staff-month

Complete Example, Embedded

- Suppose an embedded project has 50 KLOC,
  - Effort $3.6(50)^{1.20}$ = 394 staff–months
  - Development time $2.5(394)^{0.32}$ = 17 months
  - Average staff 394 / 17 = 23 staff
  - Productivity 50,000 LOC / 394 staff-months = 127 LOC / staff-month

Comparison

Item	Organic	Embedded
Effort (staff-months)	20	394
Development Time	8	17
Average Staff	2.5	23
Productivity	375	127

Intermediate COCOMO

$$E = a(KLOC)^{b} \times \underbrace{C}_{\text{New}}$$

- Where
  - E is the effort
  - a and b are constants (as before)
  - KLOC is thousands of lines of code
  - C is the effort adjustment factor

Cost Drivers

- Intermediate COCOMO introduces Cost Drivers
- They are used because
  - they are statistically significant to the cost of the project; and
  - they are *not* correlated to the project size (KLOC).