Unit 2

PROJECT PLANNING AND MANAGEMENT

Dr.K.Madheswari

Reference: chapter 26, ESTIMATION FOR SOFTWARE PROJECTS

Software Project Planning

The overall goal of project planning is to establish a pragmatic strategy for controlling, tracking, and monitoring a complex technical project.

Why?

So the end result gets done on time, with quality!

Project Planning Task Set-I

- Establish project scope
- Determine feasibility
- Analyze risks
  - Risk analysis is considered in detail in Chapter 25.
- Define required resources
  - Determine require human resources
  - Define reusable software resources
  - Identify environmental resources

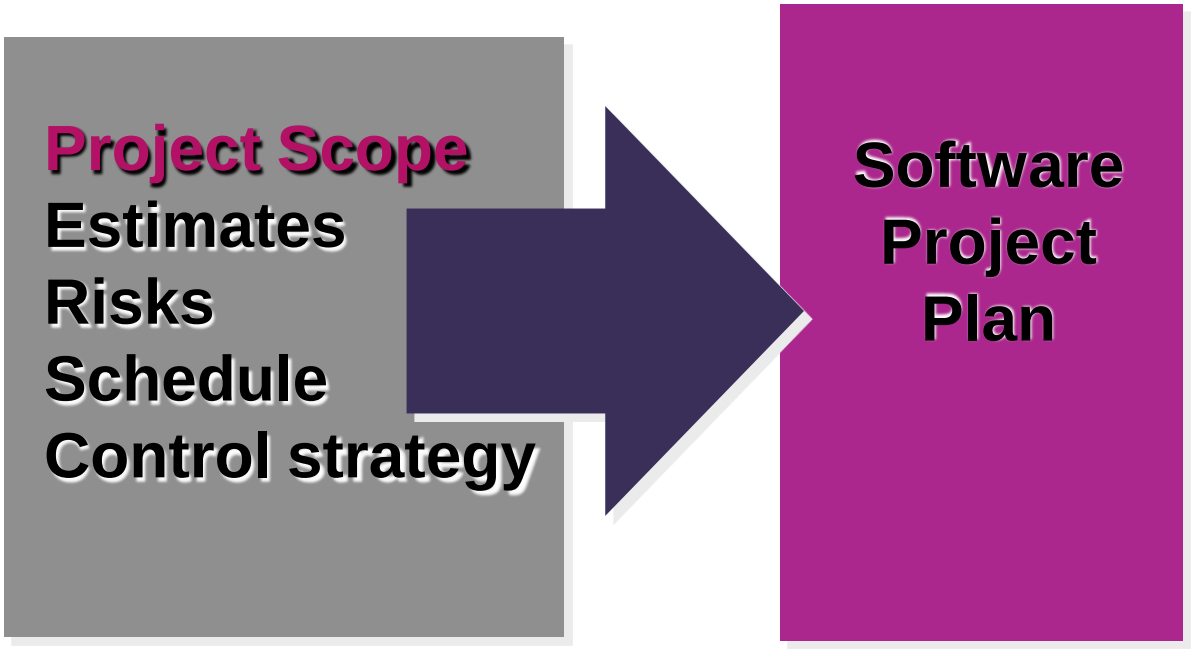
Project Planning Task Set-II

- Estimate cost and effort
  - Decompose the problem
  - Develop two or more estimates using size, function points, process tasks or use-cases
    - Reconcile the estimates
- Develop a project schedule
  - Scheduling is considered in detail in Chapter 27.
    - Establish a meaningful task set
    - Define a task network
    - Use scheduling tools to develop a timeline chart
    - Define schedule tracking mechanisms

Estimation

- Estimation of resources, cost, and schedule for a software engineering effort requires
  - experience
  - access to good historical information (metrics)
  - the courage to commit to quantitative predictions when qualitative information is all that exists
- Estimation carries inherent risk and this risk leads to uncertainty

Write it Down!



To Understand Scope ...

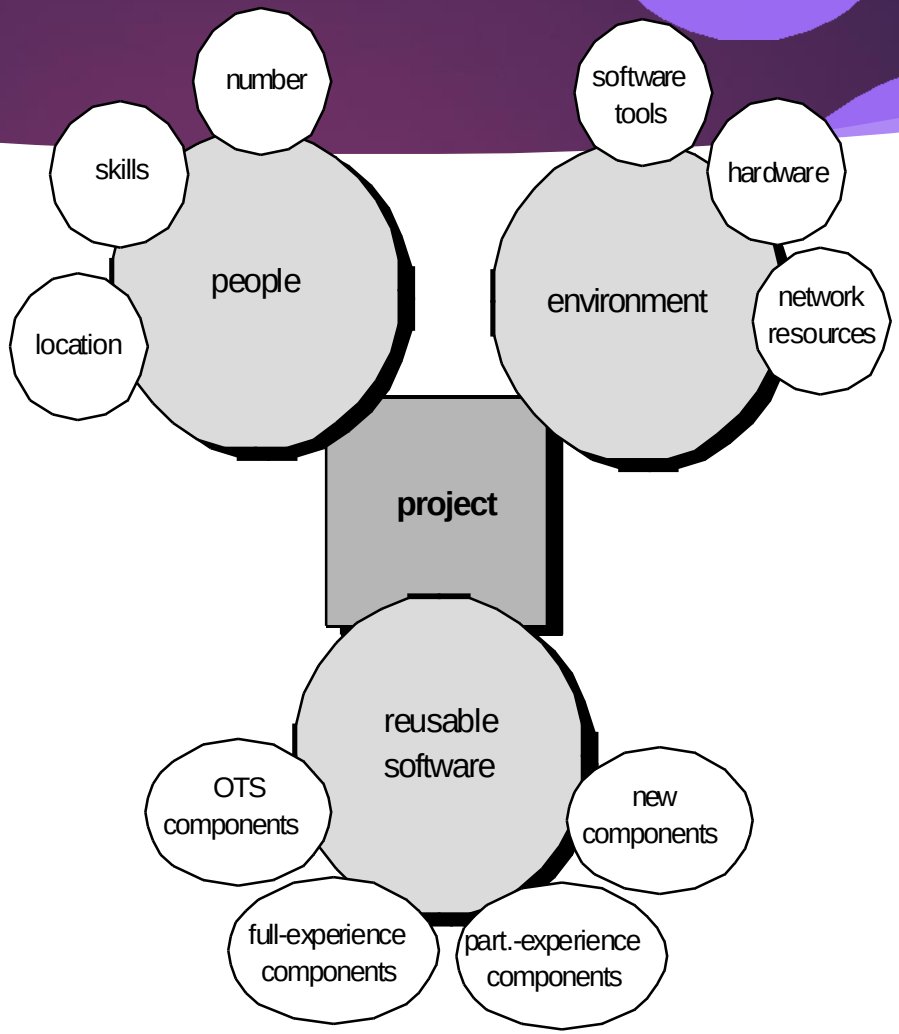
- Understand the customers needs
- understand the business context
- understand the project boundaries
- understand the customer's motivation
- understand the likely paths for change
- understand that ...

Even when you understand, nothing is guaranteed!

What is Scope?

- *Software scope* describes
  - the functions and features that are to be delivered to end-users
  - the data that are input and output
  - the “content” that is presented to users as a consequence of using the software
  - the performance, constraints, interfaces, and reliability that *bound* the system.
- Scope is defined using one of two techniques:
  - A narrative description of software scope is developed after communication with all stakeholders.
  - A set of use-cases is developed by end-users.

Resources


number
skills
people
location
software tools
hardware
environment
network resources
project
reusable software
OTS components
new components
full-experience components
part.-experience components

Four characteristics of Resources

Each resource is specified with four characteristics:

1. Description of the resource,
2. Statement of availability,
3. Time when the resource will be required, and
4. Duration of time that the resource will be applied.

The last two characteristics can be viewed as a *time window*.

Reusable Software Resources

Bennatan [Ben00] suggests ***four software resource categories*** that should be considered as planning proceeds:

Off-the-shelf components. Existing software that can be acquired from a third party or from a past project.

COTS (commercial off-the-shelf) components are purchased from a third party, are ready for use on the current project, and have been fully validated.

Full-experience components. Existing specifications, designs, code, or test data developed for past projects that are similar to the software to be built for the current project. Members of the current software team have had full experience in the application area represented by these components. Therefore, modifications required for full-experience components will be relatively low risk.

Reusable Software Resources

Partial-experience components. Existing specifications, designs, code, or test data developed for past projects that are related to the software to be built for the current project but will require substantial modification.

Members of the current software team have only limited experience in the application area represented by these components.

Therefore, modifications required for partial-experience components have a fair degree of risk.

Reusable Software Resources

New components.

Software components must be built by the software team specifically for the needs of the current project.

Environmental Resources

The environment that supports a software project, often called the *software engineering environment* (SEE), incorporates hardware and software.

Project Estimation

- Project scope must be understood
- Elaboration (decomposition) is necessary
- Historical metrics are very helpful
- At least two different techniques should be used
- Uncertainty is inherent in the process

DECOMPOSITION TECHNIQUES

Decomposition approach was discussed from two different points of view:

1. Decomposition of the problem and
2. Decomposition of the process.

Software Sizing

- The accuracy of a software project estimate is predicated on a number of things:

(1) The degree to which you have properly estimated the size of the product to be built;

(2) The ability to translate the size estimate into human effort, calendar time, and dollars (a function of the availability of reliable software metrics from past projects);

(3) The degree to which the project plan reflects the abilities of the software team;

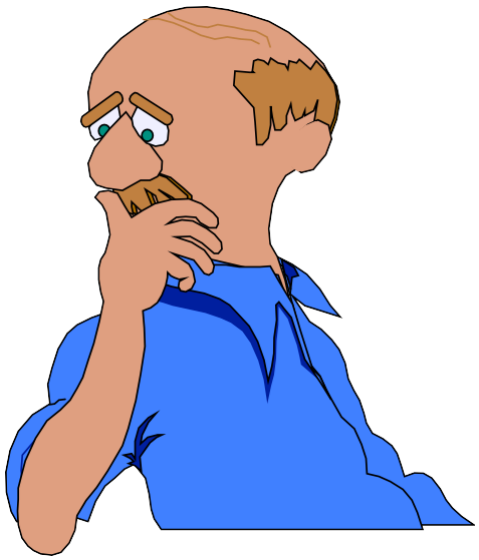
(4) The stability of product requirements and the environment that supports the software engineering effort.

How do we size the software that we're planning to build?

Putnam and Myers [Put92] suggest four different approaches to the sizing problem:

Estimation Techniques

- Past (similar) project experience
- Conventional estimation techniques
  - task breakdown and effort estimates
  - size (e.g., FP) estimates
- Empirical models
- Automated tools



Estimation Accuracy

- Predicated on …
  - the degree to which the planner has properly estimated the size of the product to be built
  - the ability to translate the size estimate into human effort, calendar time, and dollars (a function of the availability of reliable software metrics from past projects)
  - the degree to which the project plan reflects the abilities of the software team
  - the stability of product requirements and the environment that supports the software engineering effort.

Functional Decomposition

Statement of Scope

Perform a Grammatical “parse”

functional decomposition

Conventional Methods: LOC/FP Approach

- compute LOC/FP using estimates of information domain values
- use historical data to build estimates for the project

Example: LOC Approach

Function	Estimated LOC
user interface and control facilities (UICF)	2,300
two-dimensional geometric analysis (2DGA)	5,300
three-dimensional geometric analysis (3DGA)	6,800
database management (DBM)	3,350
computer graphics display facilities (CGDF)	4,950
peripheral control (PC)	2,100
design analysis modules (DAM)	8,400
estimated lines of code	33,200

Average productivity for systems of this type = 620 LOC/pm.

Burdened labor rate =$8000 per month, the cost per line of code is approximately $13.

Based on the LOC estimate and the historical productivity data, the total estimated project cost is **$431,000 and the estimated effort is 54 person-months.**

Example: FP Approach

Information Domain Value	opt.	likely	pess.	est. count	weight	FP-count
number of inputs	20	24	30	24	4	97
number of outputs	12	16	22	16	5	78
number of inquiries	16	22	28	22	5	88
number of files	4	4	5	4	10	42
number of external interfaces	2	2	3	2	7	15
count-total						321

The estimated number of FP is derived:

$$FP_{estimated} = \text{count-total } 3\ [0.65 + 0.01\ 3\ S\ (F_i)]$$

$$FP_{estimated} = 375$$

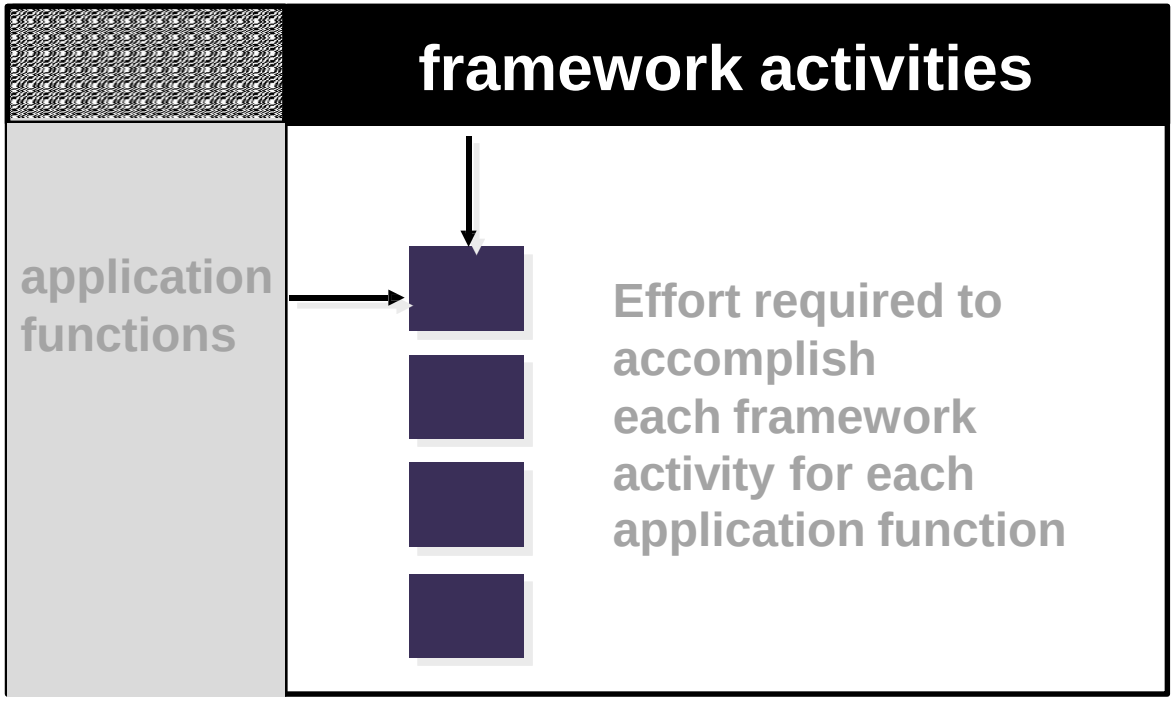
organizational average productivity = 6.5 FP/pm.

burdened labor rate = $8000 per month, approximately $1230/FP.

Based on the FP estimate and the historical productivity data, **total estimated project cost is $461,000 and estimated effort is 58 person-months.**

Process-Based Estimation

Obtained from "process framework"



Process-Based Estimation Example

Activity →	CC	Planning	Risk Analysis	Engineering		Construction Release		CE	Totals
Task →				analysis	design	code	test		
Function									
▼									
UICF				0.50	2.50	0.40	5.00	n/a	8.40
2DGA				0.75	4.00	0.60	2.00	n/a	7.35
3DGA				0.50	4.00	1.00	3.00	n/a	8.50
CGDF				0.50	3.00	1.00	1.50	n/a	6.00
DSM				0.50	3.00	0.75	1.50	n/a	5.75
PCF				0.25	2.00	0.50	1.50	n/a	4.25
DAM				0.50	2.00	0.50	2.00	n/a	5.00
Totals	0.25	0.25	0.25	3.50	20.50	4.50	16.50		46.00
% effort	1%	1%	1%	8%	45%	10%	36%		

CC = customer communication CE = customer evaluation

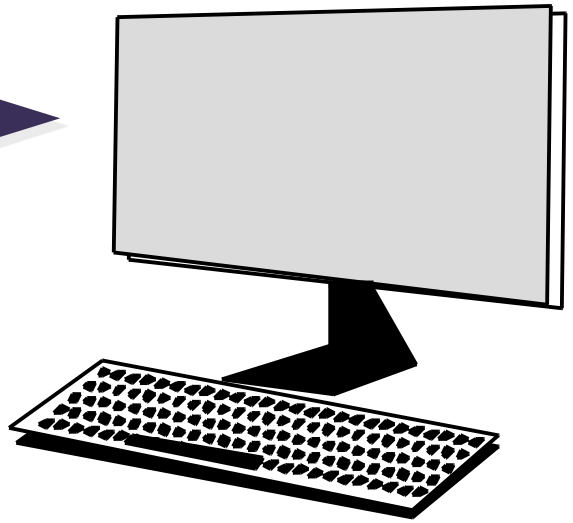
Based on an average burdened labor rate of $8,000 per month, **the total estimated project cost is $368,000 and person-months.**

Tool-Based Estimation

project characteristics

calibration factors

LOC/FP data



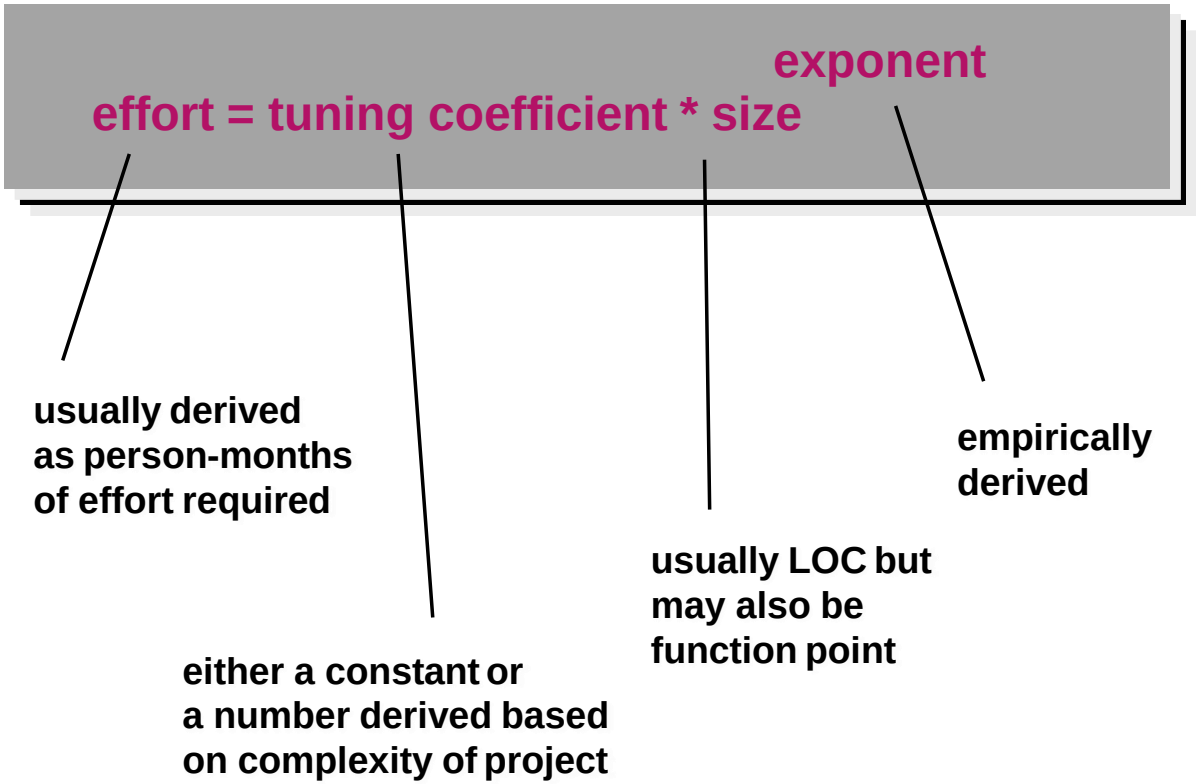
Estimation with Use-Cases

	use cases	scenarios	pages	K	scenarios	pages	LOC	LOC estimate
User interface subsystem	6	10	6	K	12	5	560	3,366
Engineering subsystem group	10	20	8	K	16	8	3100	31,233
Infrastructure subsystem group	5	6	5	K	10	6	1650	7,970
				K	K	K	K	
Total LOC estimate				K	K	K	K	42,568

Using 620 LOC/pm as the average productivity for systems of this type and a burdened labor rate of $8000 per month, the cost per line of code is approximately $13. Based on the use-case estimate and the historical productivity data, **the total estimated project cost is $552,000 and the estimated effort is 68 person-months.**

Empirical Estimation Models

General form:



COCOMO-II

- COCOMO II is actually a hierarchy of estimation models that address the following areas:
  - *Application composition model.* Used during the early stages of software engineering, when prototyping of user interfaces, consideration of software and system interaction, assessment of performance, and evaluation of technology maturity are paramount.
  - *Early design stage model.* Used once requirements have been stabilized and basic software architecture has been established.
  - *Post-architecture-stage model.* Used during the construction of the software.

The Software Equation

A dynamic multivariable model

$$E = [LOC \times B^{0.333}/P]^3 \times (1/t^4)$$

where

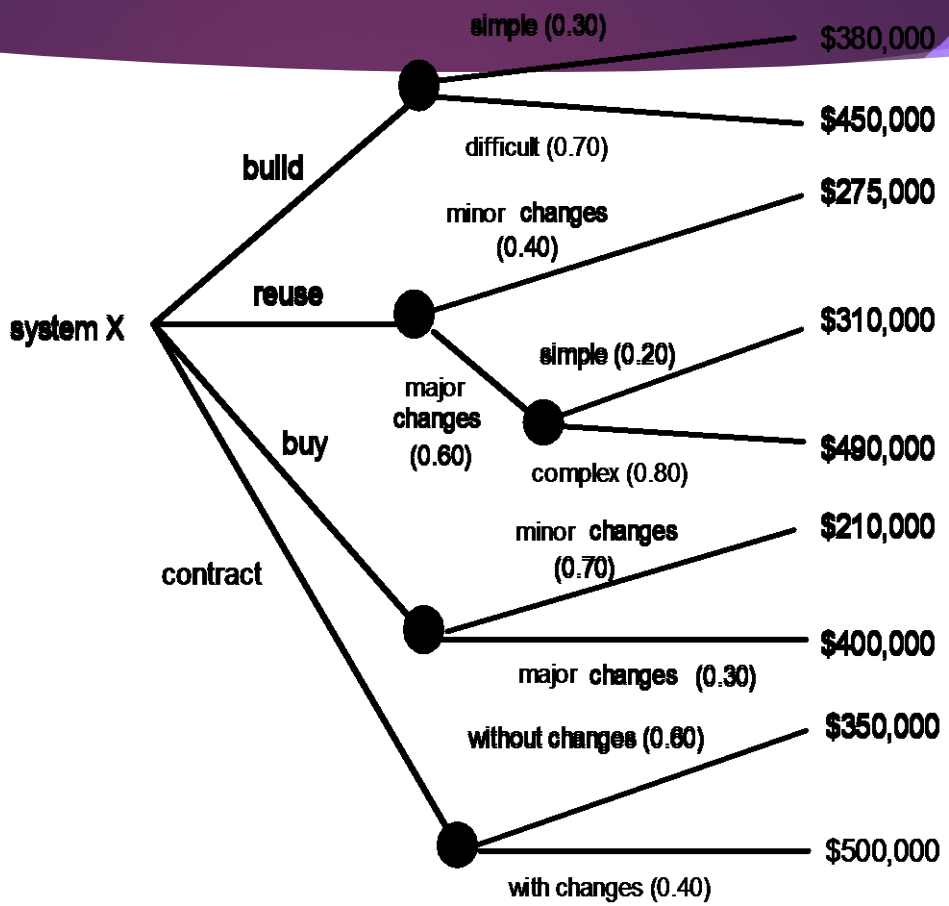
E = effort in person-months or person-years

t = project duration in months or years

B = “special skills factor”

P = “productivity parameter”

The Make-Buy Decision



Computing Expected Cost

$$\text{expected cost} = \sum_i (\text{path probability})_i \times (\text{estimated path cost})_i$$

For example, the expected cost to build is:

$$\text{expected cost}_{build} = 0.30 (\$380K) + 0.70 (\$450K)$$

$$= \$429\ K$$

similarly,

$$\text{expected cost}_{reuse} = \$382K$$

$$\text{expected cost}_{buy} = \$267K$$

$$\text{expected cost} = \$410K$$

Cont