

Lecture 17: Chapter 26

■ Estimation for Software Projects

Slide Set to accompany

Software Engineering: A Practitioner's Approach, 7/e

by Roger S. Pressman

Slides copyright © 1996, 2001, 2005, 2009 by Roger S. Pressman

For non-profit educational use only

May be reproduced ONLY for student use at the university level when used in conjunction with *Software Engineering: A Practitioner's Approach, 7/e*. Any other reproduction or use is prohibited without the express written permission of the author.

All copyright information MUST appear if these slides are posted on a website for student use.

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

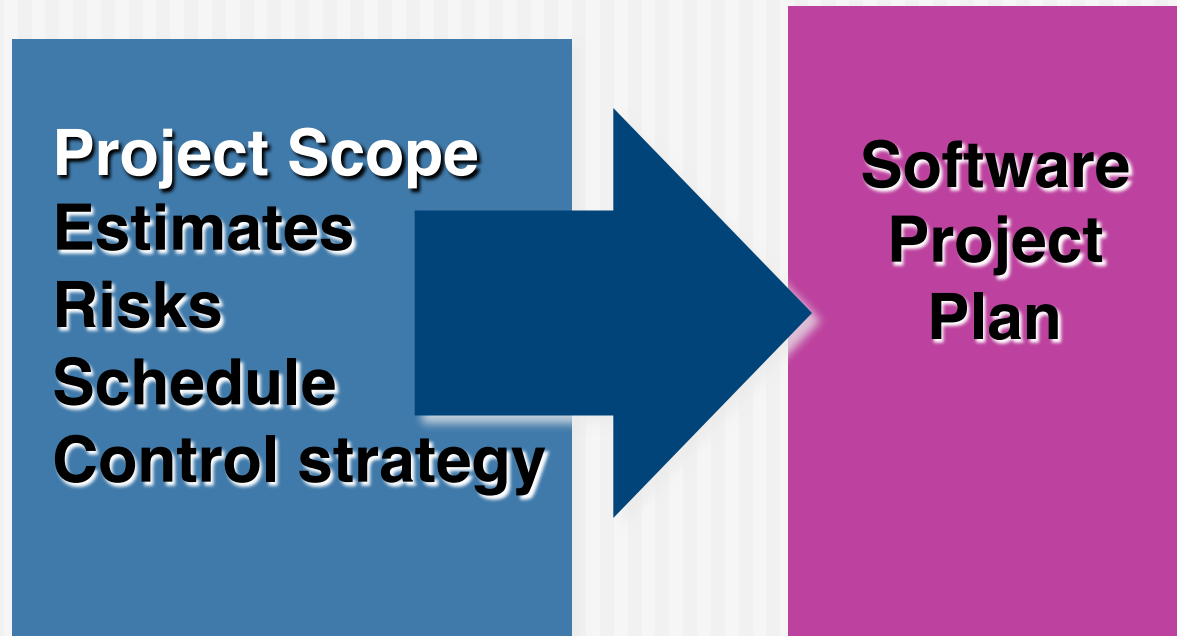
Project Planning Task Set-II

- 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 ...

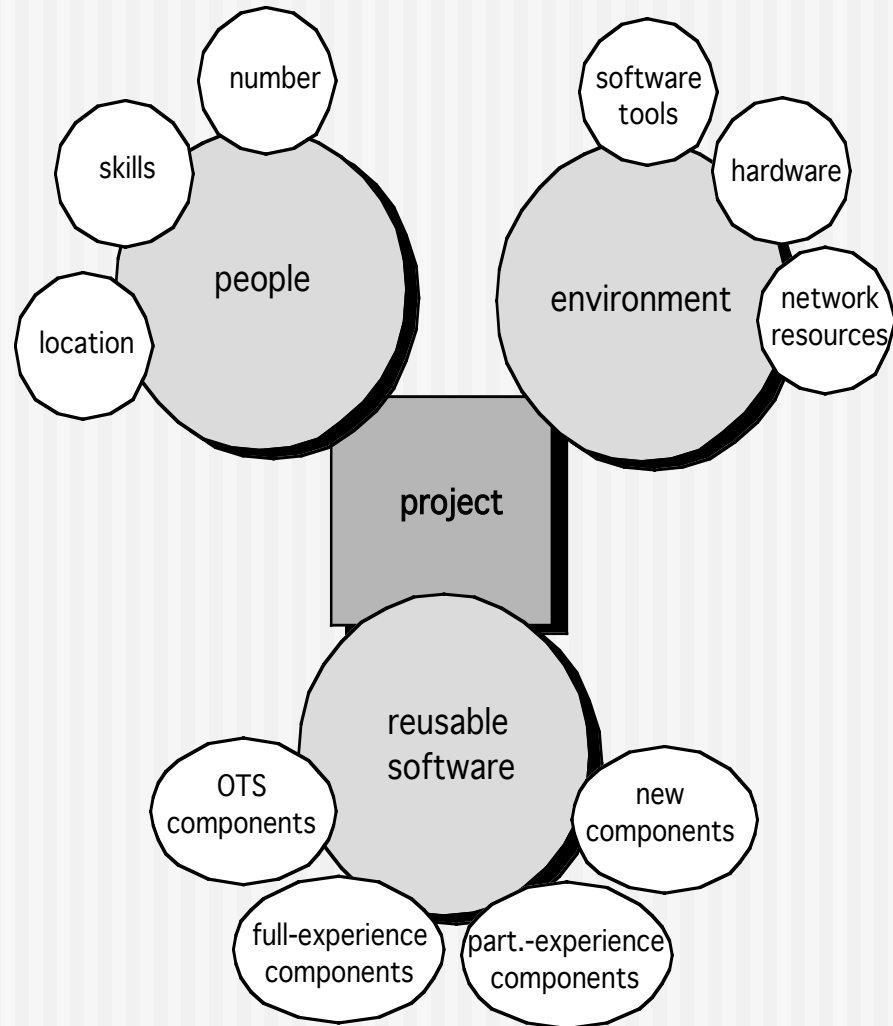
- Understand the customers needs
- understand the business context
- understand the project boundaries
- understand the customer's motivation
- understand the likely paths for change
- understand that ...

***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



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

Estimation Techniques

- Wait till the end!!
- Past (similar) project experience
- Conventional estimation techniques
 - task breakdown and effort estimates
 - size (e.g., Function Points) estimate
- 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.

Conventional Methods: LOC/FP Approach

- compute lines of code (LOC) using estimates of information domain values
- Computes function points (FP), information domain values
- use historical data to build estimates for the project
- Example: Computer-Aided Design application for mechanical components
 - Apply LOC and FP to this software

Example: LOC Approach

Function	Estimated LOC
user interface and control facilities (UICF)	2,300
two-dimensional geometric analysis (2D GA)	8,300
three-dimensional geometric analysis (3D GA)	6,800
database management (DBM)	3,380
computer graphics display facilities (CGDF)	4,980
peripheral control (PC)	2,100
design analysis modules (DAM)	8,400
<i>estimated lines of code</i>	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 $33,200 \times 620 = \text{\$431,000}$ and the estimated effort is 54 person-months.

Example: FP Approach

Information Domain Value	opt.	lkely	poss.	est. count	weight	FP-count
number of inputs	20	24	30	24	4	97
number of outputs	12	15	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:

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

$$FP_{\text{estimated}} = 375$$

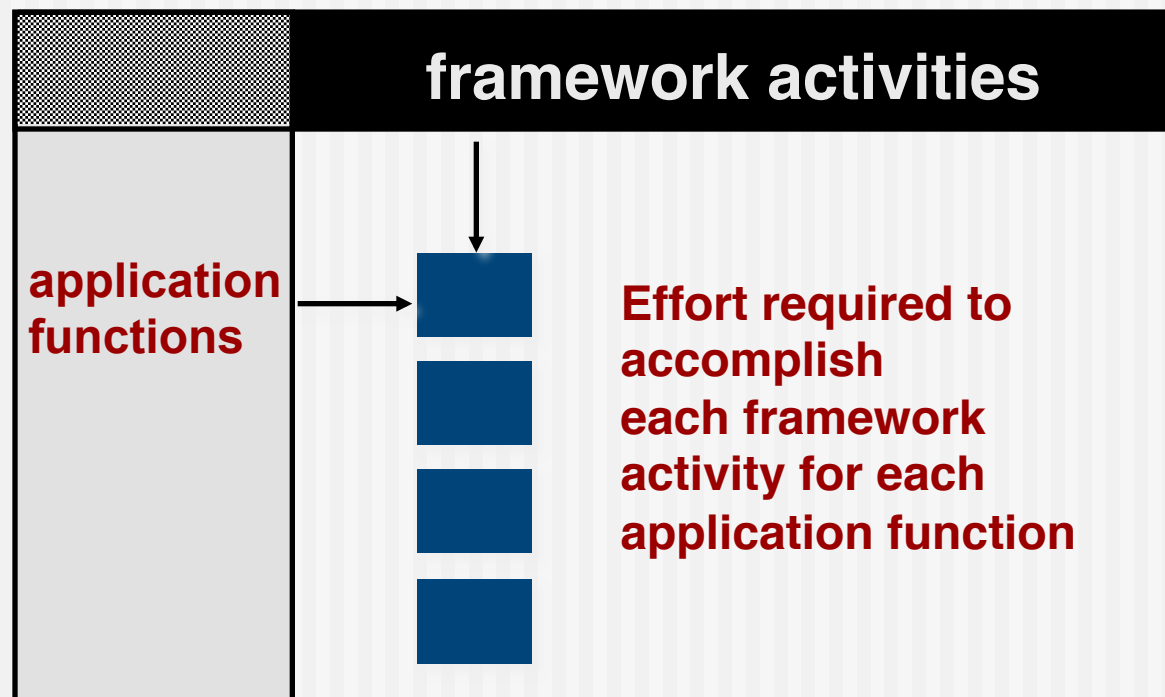
organizational average productivity = 6.5 FP/pm.

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 the estimated effort is 46 person-months.**

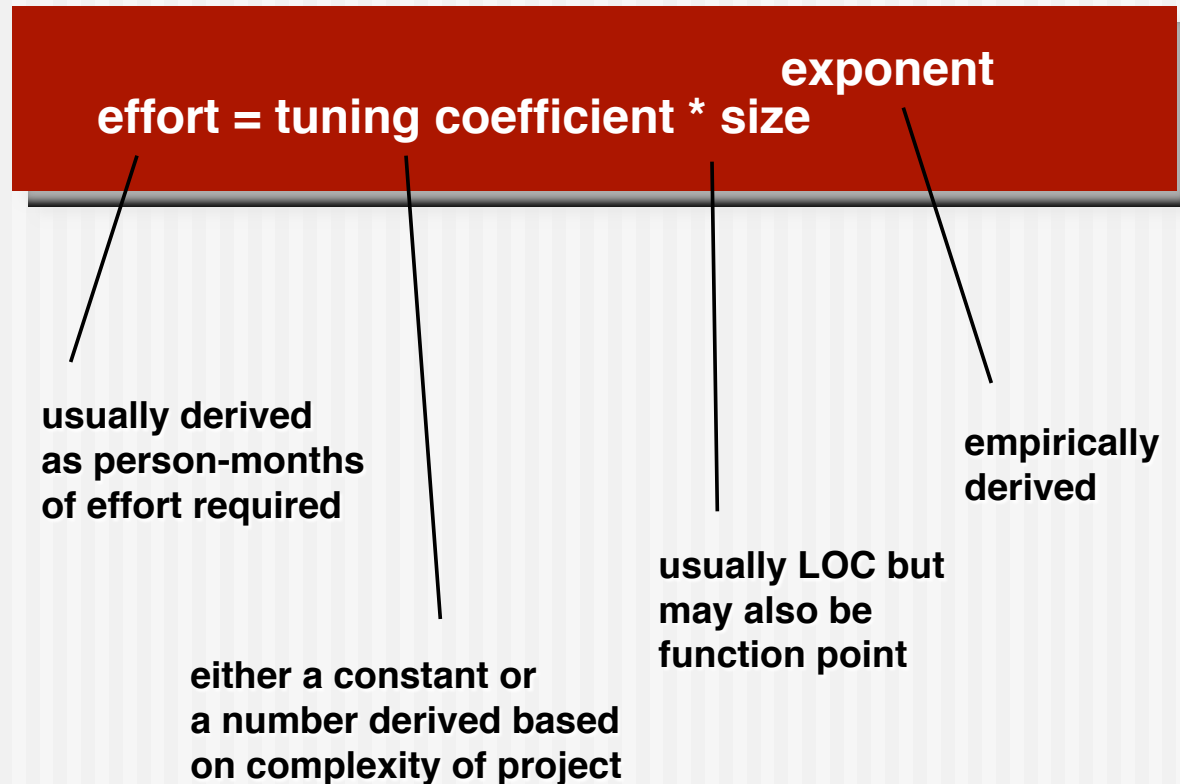
Estimation with Use-Cases

	use cases	scenarios	pages	scenarios	pages	LOC	LOC estimate
User interface subsystem	6	10	6	12	5	560	3,366
Engineering subsystem group	10	20	8	16	8	3100	31,233
Infrastructure subsystem group	5	6	5	10	6	1650	7,970
Total LOC estimate							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 \$42,568*13 = \$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.
- Empirical model that relies on size information

The Software Equation

A dynamic multivariable model

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

where

E = effort in person-months or person-years

t = project duration in months or years

B = “special skills factor”

P = “productivity parameter”

Estimation for OO Projects-I

- Develop estimates using effort decomposition, FP analysis, and any other method that is applicable for conventional applications.
- Using object-oriented requirements modeling (Chapter 6), develop use-cases and determine a count.
- From the analysis model, determine the number of key classes (called analysis classes in Chapter 6).
- Categorize the type of interface for the application and develop a multiplier for support classes:

■ Interface type	Multiplier
■ No GUI	2.0
■ Text-based user interface	2.25
■ GUI	2.5
■ Complex GUI	3.0

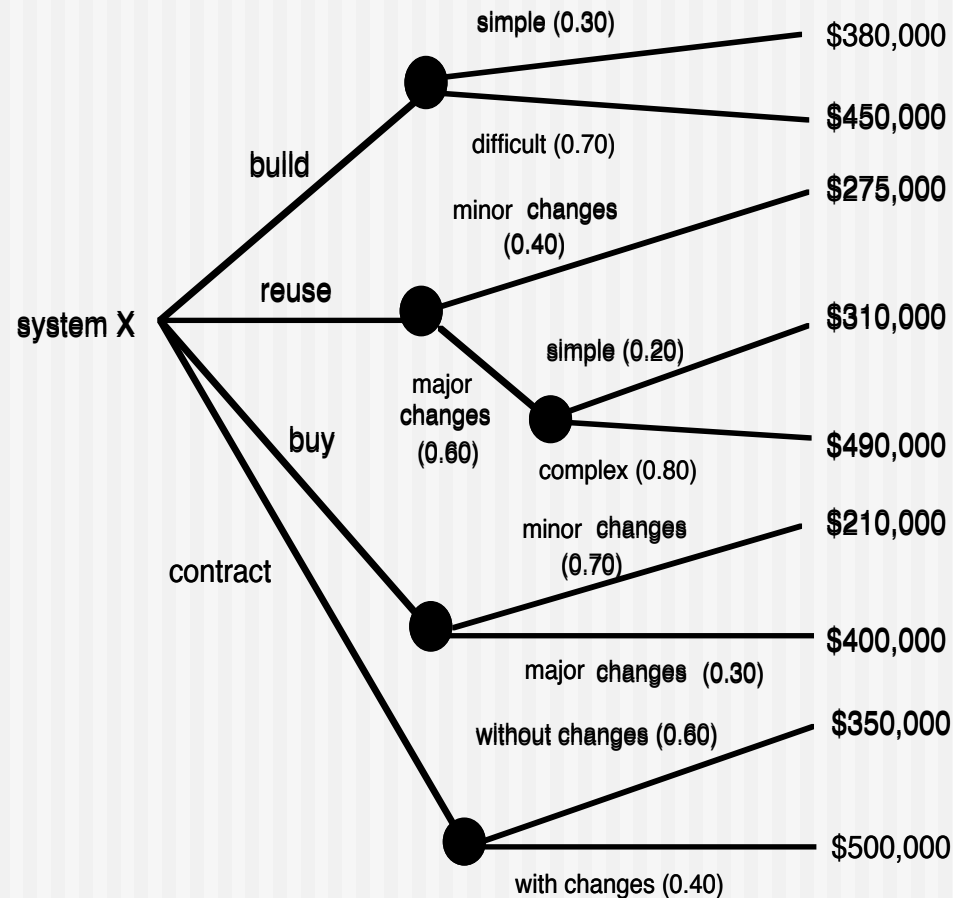
Estimation for OO Projects-II

- Multiply the number of key classes (step 3) by the multiplier to obtain an estimate for the number of support classes.
- Multiply the total number of classes (key + support) by the average number of work-units per class. Lorenz and Kidd suggest 15 to 20 person-days per class.
- Cross check the class-based estimate by multiplying the average number of work-units per use-case

Estimation for Agile Projects

- Each user scenario (a mini-use-case) is considered separately for estimation purposes.
- The scenario is decomposed into the set of software engineering tasks that will be required to develop it.
- Each task is estimated separately. Note: estimation can be based on historical data, an empirical model, or “experience.”
 - Alternatively, the ‘volume’ of the scenario can be estimated in LOC, FP or some other volume-oriented measure (e.g., use-case count).
- Estimates for each task are summed to create an estimate for the scenario.
 - Alternatively, the volume estimate for the scenario is translated into effort using historical data.
- The effort estimates for all scenarios that are to be implemented for a given software increment are summed to develop the effort estimate for the increment.

The Make-Buy Decision



Computing Expected Cost

expected cost =

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

For example, the expected cost to build is:

$$\begin{aligned}\text{expected cost}_{\text{build}} &= 0.30 (\$380\text{K}) + 0.70 (\$450\text{K}) \\ &= \$429 \text{ K}\end{aligned}$$

similarly,

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

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

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