Chapter 33

■ Estimation for Software Projects

Slide Set to accompany

Software Engineering: A Practitioner's Approach, 8/e

by Roger S. Pressman

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 35.
- 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

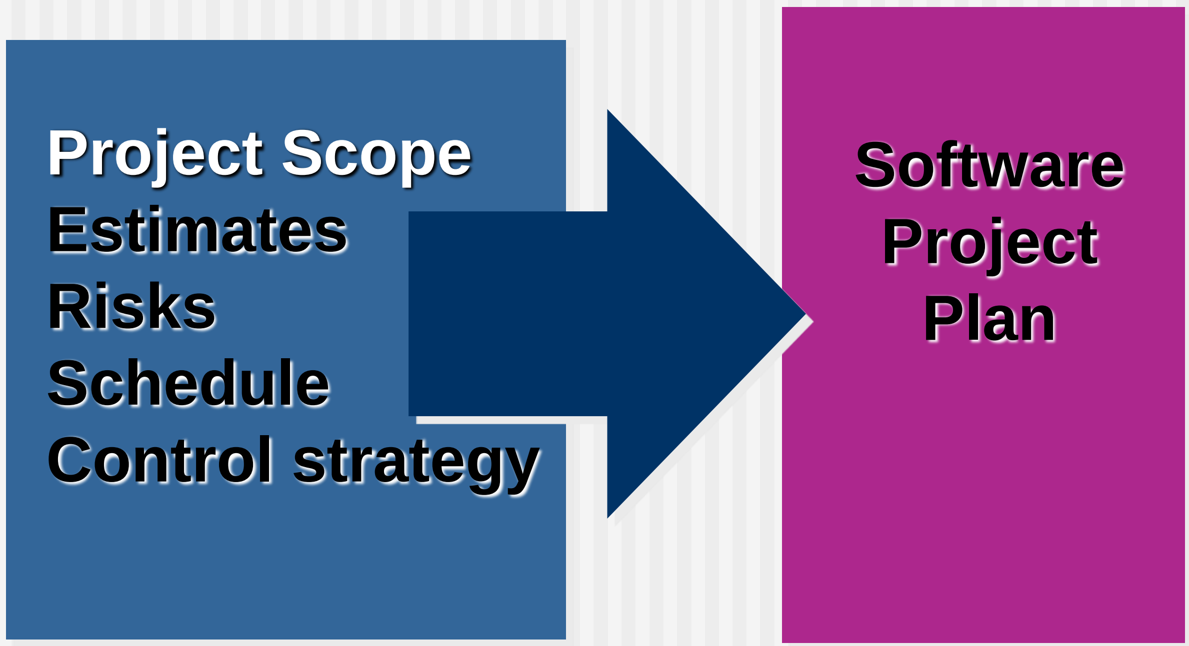
Estimation

- 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

Estimation

- estimation risk is influenced by:
  - Project complexity
  - *Project size*
  - *degree of structural uncertainty*
  - The availability of historical information

Write it Down!



To Understand Scope ...

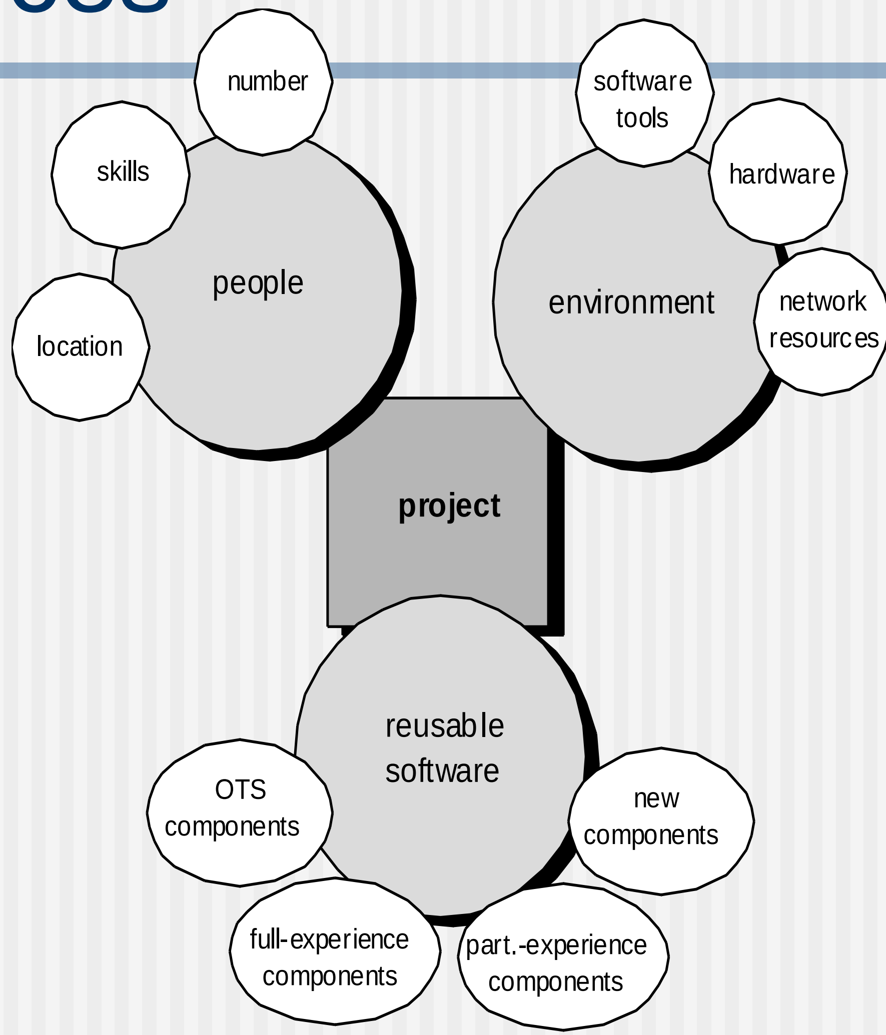
- Understand the customers needs
- understand the business context
- understand the project boundaries
- understand the customer’s motivation
- understand the likely paths for change
- understand that ...

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

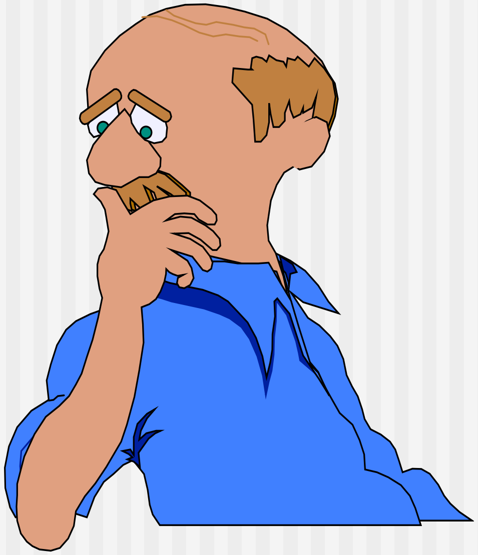

number
skills
people
location
software tools
hardware
environment
network resources
project
reusable software
OTS components
new components
full-experience components
part.-experience components

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

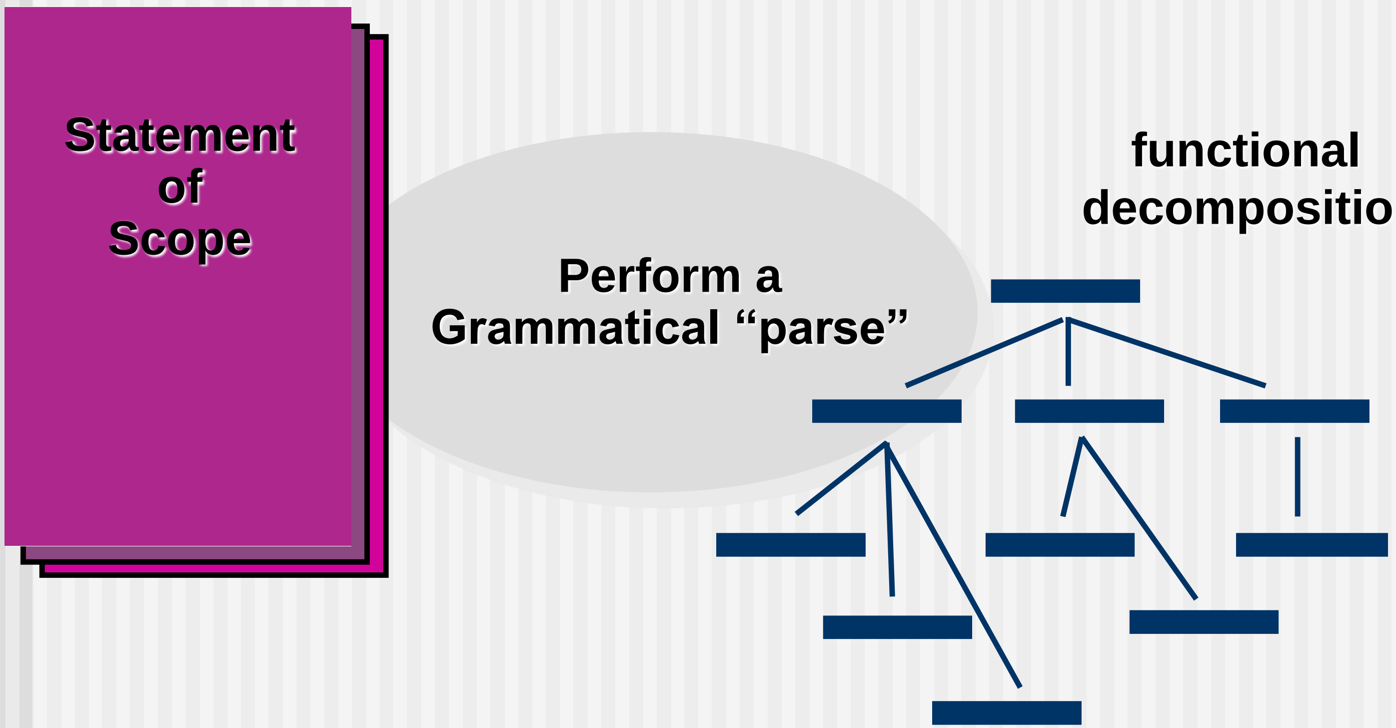
- Past (similar) project experience
- Conventional estimation techniques
  - task breakdown and effort estimates
  - size (e.g., FP) estimates
- Empirical models
- Automated tools



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



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

The mechanical CAD software will accept two- and three-dimensional geometric data from a designer. The designer will interact and control the CAD system through a user interface that will exhibit characteristics of good human/machine interface design. All geometric data and other supporting information will be maintained in a CAD database. Design analysis modules will be developed to produce the required output, which will be displayed on a variety of devices. The software will be designed to control and interact with peripheral devices that include a mouse, scanner, laser printer, and plotter.

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 function (PCF)	2,100
Design analysis modules (DAM)	8,400
Estimated lines of code	*33,200*

$$\frac{4600 + 4(6900) + 8600}{6} = 6800$$

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 external inputs	20	24	30	24	4	97
Number of external outputs	12	15	22	16	5	78
Number of external inquiries	16	22	28	22	5	88
Number of internal logical files	4	4	5	4	10	42
Number of external interface files	2	2	3	2	7	15
Count total						*320*

The estimated number of FP is derived:

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

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

organizational average productivity = 6.5 FP/pm.

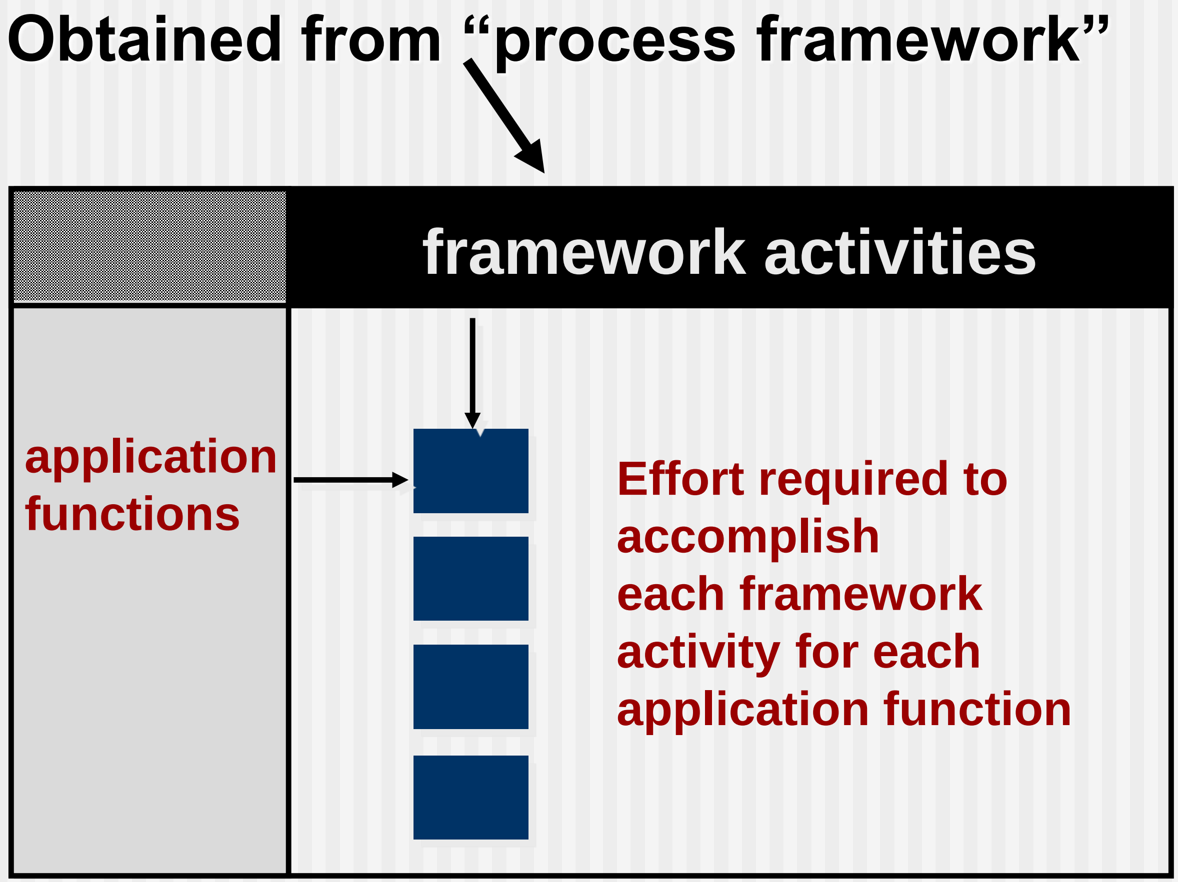
burdened labor rate = \$8000 per month, approximately \$1230/FP.

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

Example: FP Approach

Factor	Value
Backup and recovery	4
Data communications	2
Distributed processing	0
Performance critical	4
Existing operating environment	3
Online data entry	4
Input transaction over multiple screens	5
Master files updated online	3
Information domain values complex	5
Internal processing complex	5
Code designed for reuse	4
Conversion/installation in design	3
Multiple installations	5
Application designed for change	5
Value adjustment factor	**1.17**

Process-Based Estimation



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

unadjusted use case weight (UUCW)

Use Case Type	Description	Factor
Simple	3 or fewer transactions	5
Medium	4–7 transactions	10
Complex	>7 transactions	15

Estimation with Use-Cases

unadjusted actor weight (UAW)

Use Case Type	Description	Factor
Simple	Simple actors are automatons (another system, a machine or device) that communicate through an API.	1
Medium	Average actors are automatons that communicate through a protocol or a data store	2
Complex	humans who communicate through a GUI or other human interface	3

Estimation with Use-Cases

technical complexity factors (TCFs)

	Factor	Weight
1	Distributed system	2
2	Response or throughput performance objectives	1
3	End-user efficiency (online)	1
4	Complex internal processing	1
5	Code must be reusable	1
6	Easy to install	0.5
7	Easy to use	0.5
8	Portable	2
9	Easy to change	1
10	Concurrent	1
11	Includes special security features	1
12	Provides direct access for third parties	1
13	Special user training facilities required	1
TFactor		$\sum_{i=1}^{i=13}$

$$TCF = 0.6 + (0.01 * \text{TFactor})$$

Estimation with Use-Cases

environment complexity factors (ECFs)

	Factor	Weight
1	Familiar with Internet process	1.5
2	Application experience	0.5
3	Object-oriented experience	1
4	Lead analyst capability	0.5
5	Motivation	1
6	Stable requirements	2
7	Part-time workers	–1
8	Difficult programming language	–1
EFactor		$\sum_{i=1}^{i=8}$

$$ECF = 1.4 + (-0.03 * EFactor)$$

Estimation with Use-Cases

- Use case point

$$UCP = (UUCW + UAW) \times TCF \times ECF$$

Estimation with Use-Cases

- Use case point

$$\text{UUCW} = (16 \text{ use cases} \times 15) + [(14 \text{ use cases} \times 10) + (8 \text{ use cases} \times 5)] + (10 \text{ use cases} \times 5) = 470$$

$$\text{UAW} = (8 \text{ actors} \times 1) + (12 \text{ actors} \times 2) + 4 \text{ actors} \times 3) = 44$$

$$\text{TCF} = 1.04$$

$$\text{ECF} = 0.96$$

$$\text{UCP} = (470 + 44) \times 1.04 \times 0.96 = 513$$

Using 620 LOC/pm as the average productivity for systems of this type and a burdened labor rate of $8,000 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 about 70 person-months**

Empirical Estimation Models

General form:

$$\text{effort} = A + B * (e_v)^C$$

usually derived as person-months of effort required

A, B, and *C* are empirically derived constants

e_v

is the estimation variable (either LOC or FP)

Empirical Estimation Models

$E = 5.2 \times (\text{KLOC})^{0.91}$	Walston-Felix model
$E = 5.5 + 0.73 \times (\text{KLOC})^{1.16}$	Bailey-Basili model
$E = 3.2 \times (\text{KLOC})^{1.05}$	Boehm simple model
$E = 5.288 \times (\text{KLOC})^{1.047}$	Doty model for KLOC > 9

FP-oriented models have also been proposed. These include

$E = -91.4 + 0.355\ \text{FP}$	Albrecht and Gaffney model
$E = -37 + 0.96\ \text{FP}$	Kemerer model
$E = -12.88 + 0.405\ \text{FP}$	Small project regression model

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.

COCOMO-II

Object type	Complexity weight		
	Simple	Medium	Difficult
Screen	1	2	3
Report	2	5	8
3GL component			10

$$\text{NOP} = (\text{object points}) \times [(100 - \%\text{reuse})/100]$$

$$\text{PROD} = \frac{\text{NOP}}{\text{person-month}}$$

$$\text{Estimated effort} = \frac{\text{NOP}}{\text{PROD}}$$

Developer's experience/capability	Very low	Low	Nominal	High	Very high
Environment maturity/capability	Very low	Low	Nominal	High	Very high
PROD	4	7	13	25	50

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 Software Equation

A dynamic multivariable model

$$E = \frac{\text{LOC} \times B^{0.333}}{P^3} \times \frac{1}{t^4}$$

where

- E = effort in person-months or person-years
- t = project duration in months or years
- B = “special skills factor”

  B increases slowly as “the need for integration, testing, quality assurance, documentation, and management skills grows” .For small programs (KLOC 5 5 to 15), *B* 5 0.16. For programs greater than 70 KLOC, *B* = 0.39.
- P = “productivity parameter”

The Software Equation

$$t_{min} = 8.14 * \left({}^{LOC}/_{p}\right)^{0.43} \text{ in months for } t_{min} > 6 \text{ months}$$

$$E = 180\, Bt^{3} \text{ in person-months for } E \geq 20 \text{ person-months}$$

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

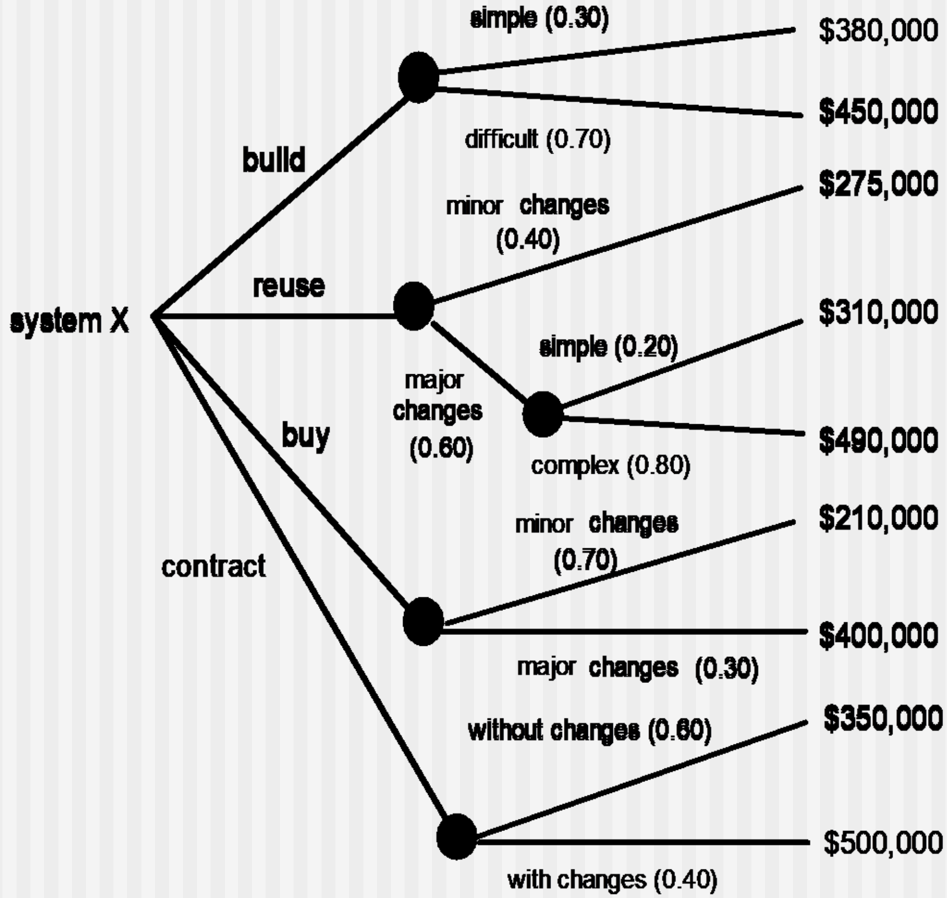
Estimation for OO Projects-II

- 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 =

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

For example, the expected cost to build is:

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

$$= \$429\ \text{K}$$

similarly,

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

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

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