

Cost Estimation Strategies

- Algorithmic models (Rayleigh curve Cost in week $t = K a t \exp(-a t^2)$)
- Expert judgment (9 step model presented later)
- Analogy (Use similar systems)
- Parkinson (Work expands to fill the available budget)*
- Price-to-win (A political or economic decision)*
- Top Down (Use global properties of software)
- Bottom Up (Estimate each component)

* - **Not based on any technical foundation**

COST ESTIMATION GUIDELINES

- These guidelines are by Donald J. Reifer
- Based upon work by J. D. Aron, "Estimating resources for large programming systems," Software Engineering: Concepts and Techniques, Litton Educational Services, 1976.
- Applicable to projects over 10,000 lines and staff of 10.
- Defined by 9 step process.
- Although developed in 1970s, most rules still applicable

Step 1: Define work element

Develop work breakdown structure (WBS)

- What activities are within scope of this work element?
- What programs are included within this element?
- What are documentation requirements?
- What are operational support requirements?
- How volatile are requirements?
- Do I understand the nature of job to estimate these?
- Do I need help and where can I get it?

MSWE 609

3

Typical WBS tasks

Typical activities:

- Software systems engineering (requirements analysis, interface analysis)
- Software development (Software life cycle- requirements, architectural design, procedural design, code, unit test, integration test, technical reviews, training; COTS -Requirements, package evaluation, acquisition, installation)
- Software test and evaluation (development test, acceptance test, test bed and tool support, test data management, test reviews)
- Management (project management, administrative support, management tools, management reviews, management training)
- Product assurance (configuration management, Library operations, interface control, data management, quality assurance, quality control)
- Operations and support (maintenance, support management, test and operational support)

MSWE 609

4

Step 2: Determine software size

Methods from last time:

- Top-down estimating
- Similarities and differences estimating
- Ratio estimating
- Standards estimating
- Bottom-up estimating

More about size measures next week

MSWE 609

5

Step 3: Assess difficulty

Difficulty Category	Characteristics	Examples
Easy	<ul style="list-style-type: none">• Few interactions with other system elements• Simple algorithm• Previous experience• Data-driven architecture	<ul style="list-style-type: none">• Payroll systems• Report generation systems• Hardware diagnostics
Medium	<ul style="list-style-type: none">• Some interactions• Suitable algorithms exist• Have previous experience in application• New machine or language	<ul style="list-style-type: none">• Numerical analysis• Production control• Management information system
Hard	<ul style="list-style-type: none">• Many interactions• Algorithms are new• Requirements are volatile• Limited experience in application• Message-driven architecture	<ul style="list-style-type: none">• On-line systems• Real-time software• Systems software• Message switching systems

MSWE 609

6

Step 4: Define risk and its impact

Risk item	Risk reduction	Cost impact
Late delivery of hardware	Acquire time on another system	Computer time costs
New operating system	Benchmark early and perform acceptance test before use	Staffing to prepare benchmark
Feasibility of requirements	Feasibility analysis and simulation	Staffing to conduct analysis
Staffing up	Start recruiting and training early	Recruiting and training costs
Feasibility of design	Peer reviews	Added effort for review preparation
Lack of management visibility	Use detailed work packaging and weekly reporting	Added effort to prepare inputs for reports
Configuration integrity of software products	Use formal change control system	Purchase price of library plus administrative costs
Lack of a test discipline	Use independent test group and get them involved early	Cost of using test group

MSWE 609

7

Additional risk factors

From Capers Jones:

1. Inaccurate metrics
2. Inadequate measurement
3. Excessive schedule pressure
4. Inaccurate cost estimating
5. Management malpractice (e.g., knowledge)
6. Silver bullet syndrome (will save the project)
7. Creeping user requirements
8. Low quality
9. Low productivity
10. Cancelled projects

From Barry Boehm:

1. Personnel shortfalls
2. Unrealistic schedules and budgets
3. Wrong functionality
4. Wrong user interface
5. Gold plating
6. Continuous requirements changes
7. Shortfalls in externally supplied components
8. Shortfalls in externally provided tasks
9. Real time performance constraints
10. Straining computer science technology

MSWE 609

8

Risk: Some risk is subjective

Expected value = Prob(success)*Payoff(success) + Prob(failure)*Payoff(failure)

You can win between 0 and \$1000

1. Spin a dial containing the numbers 1 through 10.

Choose A or B:

A: If 1—9 show up, you get nothing; if 10 shows up you get \$1000.

B: Win \$100 for doing nothing

Most people choose B as “found money.” (E.g., in MSWE607 class 31 out of 46 chose this option.) It didn’t matter that they had the potential for winning \$1000.

You can lose between 0 and \$1000

2. Spin a dial containing the numbers 1 through 10.

Choose A or B:

A: If 1—9 show up, you pay nothing; if 10 shows up you pay \$1000.

B: Pay \$100 for doing nothing

Most people chose A, even though they risked losing \$1000 (e.g., only 16 out of 46 in MSWE 607 chose paying \$100).

But both experiments were the same. Each had an expected value (average gain or loss) of \$100. Yet on the chances of losing money, people were more willing to gamble than if they were guaranteed a win.

MSWE 609

9

Risk: Utility functions

Given two choices with different payoffs, what amount should you receive so that you are indifferent as to which choice to choose? For example, in the previous example:

Spin a dial containing the numbers 1 through 10.

Choose A or B:

A: If 1—9 show up, you get nothing; if 10 shows up you get \$1000.

B: Win \$ x for doing nothing

\$100 is the expected value of the payoff.

What is the value of x so that you are indifferent to choice A and B?

- If you “take risks,” then you will choose A for fairly high values of B . Only if B is extremely high will you choose B. For \$100 you would rather choose A.
- If you are “risk adverse”, you will choose the “sure thing.” In this case, you will choose B even for low values of x less than \$100.

Everyone has a breakeven point. This gives a model where we can measure the value of this risk.

MSWE 609

10

Step 5: Estimate personnel resources

- $Cost = Size * Dollars/Line$
(Caveat: salaries are 1976)

- If reused software:
 $Cost = size(new) * \$/Line$
 $+ size(Reused) * \$/Line * (Relative\ effort\ factor)$

Note: SEL uses for reused software:
 $Size = LinesNewCode$
 $+ .2 * LinesReusedCode$

Use these figures for \$/Line

Duration \ Difficulty	< 1 year	1-2 years	2-3 years	> 3 years
Easy	\$40	\$30	\$25	\$40
Medium	\$50	\$40	\$35	\$50
Hard	\$75	\$50	\$45	\$65

Note: Salary figures are 20 years out of date

Step 6: Adjust estimate

Adjust for:

- language (assembly code or HLL [divide by 2]),
- capacity (e.g., time or space limited [multiply by 3 % limited]),
- methodology (divide by 1.5),
- risk (Add in cost of risk -covered in later slides)

This is the major step that has changed since 1976

Step 7-9: Remaining steps

Step 7: Extrapolate estimate: Apply estimate to all work elements of step 1

Step 8: Schedule effort: Create PERT chart to schedule resources

Step 9: Reiterate: Pose following questions are reiterate:

- Can we do it in half the time?
- Can we do it for half the money?
- Can we do more for the same money?

MSWE 609

13

COCOMO – COst Containment Model

- **Boehm – 1981.** The assumption is that effort is related to size as follows: $\text{Effort} = k(\text{LOC})^p$ where k and p depend upon application domain and environment and $p > 1$.
- **Much of COCOMO is based upon 1970s technology which underlies many of the assumptions in the model (e.g., batch environments, DoD contracting, FORTRAN and similar languages).**
- **One of the weaknesses of COCOMO is that other environments may not have the same underlying assumptions used in setting the parameters.**
- **Around 1995 or 1996 Boehm started to work on a revision to COCOMO called COCOMO-2. In general it does not do as well as COCOMO in average error in estimating effort.**

MSWE 609

14

1965 SDC model

Effort (months) = -33.63

- + 9.15 (lack of requirements)(0-2)
- + 10.73 (Stability of design (0-3)
- + 0.51 (Percent math)
- + 0.46 (Percent storage/retrieval instructions)
- + 0.40 (Number of subprograms)
- + 7.28 (Programming language (0-1)
- 21.45 (Business applications) (0-1)
- + 13.53 (Stand alone program (0-1)
- + 12.35 (First program on computer (0-1)
- + 58.82 (Concurrent hardware development (0-1)
- + 30.61 (Random access device used (0-1)
- + 29.55 (Different host target (0-1)
- + 0.54 (Number of personnel trips)
- 25.20 (Developed by military organization (0-1)

MSWE 609

15

Nominal effort

Basic effort equations:

- Organic development (well understood)
 $MM = 3.2 \text{ KLOC}^{1.05}$
- Semidetached (More complex)
 $MM = 3.0 \text{ KLOC}^{1.12}$
- Embedded (highly complex)
 $MM = 2.8 \text{ KLOC}^{1.20}$

Then multiply MM by 15 COCOMO factors

MSWE 609

16

COCOMO factors

Cost Driver	Low value	High value
Product attributes		
Required reliability	.75	1.40
Data base size	.94	1.16
Product complexity	.70	1.65
Computer attributes		
Execution constraints	1.00	1.66
Main storage constraints	1.00	1.58
Virtual machine volatility	.87	1.30
Computer turnaround time	.87	1.15
Personnel attributes		
Analyst capabilities	1.46	.71
Application experience	1.29	.82
Programmer capability	1.42	.70
Virtual machine experience	1.21	.90
Programming language experience	1.14	.95
Project attributes		
Methodology	1.24	.82
Software tools	1.24	.83
Required development schedule	1.23	1.10

MSWE 609

17

COCOMO II

COCOMO/SCM 14 -- COCOMO development is continuing*

Theme: Software Cost, Schedule, and Process, October 26-28, 1999

Last year's Forum introduced a number of emerging extensions to the COCOMO II estimation model. These included COCOTS for COTS integration, CORADMO for Rapid Application Development, COQUALMO for quality estimation, and COSSEMO for stagewise schedule and effort estimation. Discussions at the Forum indicated that these and other cost and schedule estimation models need to address the effects of process selection on cost, schedule, and quality. The choice of process objectives (optimize cost, schedule, or quality) and process architectures (waterfall, incremental, evolutionary, spiral) can have significant effects on a product's overall cost, schedule, and quality, and on the distribution of cost and schedule bystage.

This year's Forum particularly solicits presentations addressing these cost/schedule/process issues, either via experience reports or modeling and data analysis. Besides these topics, the Forum also solicits papers on other software cost estimation-related topics such as cost models for new software approaches (object-oriented, COTS integration, software product lines); software sizing; and relations between software cost estimation and software risk assessment, requirements, architecture, value assessment, and quality assessment.

* - Email announcement, September 13, 1999

MSWE 609

18