Criteria for Project Evaluation (Cost, Risk)

Saeed Siddik
Assistant Professor
IIT University of Dhaka

Cost

Cost Evaluation & Estimation

- **Why Cost Estimation Matters**
- Role in budgeting, staffing, planning, and decision-making.
- Consequences of underestimation and overestimation.

Software Cost Components

- Direct costs (labor, tools, materials)
- Indirect costs (training, overheads)
- Hidden costs (rework, delays, integration issues)

Cost Estimation Techniques

- Expert judgment
- Algorithmic models (COCOMO: Constructive Cost Model)
- Analogy-based estimation
- Parametric models
- Use-case-based estimation
- Machine learning methods in recent research.

Example: Software Project Cost Estimation

Project Scenario:

A company plans to develop a new payroll software system. The project has 3 main modules. Here I listed the Lines of Code (LOC) and Complexity below:

- M1: Employee Management(loc: 5,000; Medium)
- M2: Payroll Calculation(loc: 8,000; High)
- M3: Reporting & Dashboard (loc: 3,000; Low)

- Average cost per developer-day = BDT 400

Cost: Expert Judgment Estimation

- A senior project manager estimates that each module will require:
  - M1 = 120 developer-days = 120×400 → BDT 48,000
  - M2 = 200 developer-days = 200×400 → BDT 80,000
  - M3 = 60 developer-days = 60×400→ BDT 24,000
  - **Total Cost** = BDT 48,000 + BDT 80,000 + BDT 24,000 = BDT 152,000

- **Interpretation:** Relies on expert knowledge and experience. Simple but subjective.

Cost: Analogy-Based Estimation

- If a similar previous project with 12,000 LOC cost BDT 180,000, we can scale:
- Total LOC for current project =
  - 5,000 + 8,000 + 3,000 = 16,000 LOC
- Cost per LOC from previous project =
  - BDT 180,000 / 12,000 = BDT 15/LOC
- **Estimated Cost** = 16,000 × BDT 15 = BDT 240,000

- **Interpretation**: Uses historical data. Quick but depends on similarity.

Cost: Bottom-Up Estimation

Estimate cost for each module individually based on LOC and productivity:

Suppose,
Average productivity (LOC per developer-day):

Low: 50 LOC/day
Medium: 40 LOC/day
High: 30 LOC/day

Cost: Bottom-Up Estimation

Estimate cost for each module individually based on LOC and productivity:

Module	LOC	Complexity	Productivity (LOC/day)	Developer-Days	Cost ($)
M1	5,000	Medium	40	$5000/40 = 125$	$125 \times 400 = 50{,}000$
M2	8,000	High	30	$8000/30 \approx 267$	$267 \times 400 \approx 106{,}800$
M3	3,000	Low	50	$3000/50 = 60$	$60 \times 400 = 24{,}000$

Total Cost = $50{,}000 + 106{,}800 + 24{,}000 = \$180{,}800$

Interpretation: Detailed and systematic. Most accurate if data is reliable.

COCOMO (Constructive Cost Model) – Basic Estimation

- COCOMO formula (Organic Mode):
  - $\text{Effort(person−months)} = a \times (\text{KLOC})^{b}$

- For **organic projects**, typical constants: a = 2.4, b = 1.05

- KLOC = total lines of code / 1000 = 16,000 / 1000 = 16

- $\text{Effort} = 2.4 \times (16)^{1.05} \approx 2.4 \times 17.4$
  ≈ 41.8 person-months

Constraints in COCOMO

- COCOMO defines three project modes based on complexity, size, and team experience:

Mode	Description	a	b
Organic	Small, simple, familiar projects; clearly understand requirements and technology	2.4	1.05
Semi-Detached	Medium-sized, mixed complexity	3	1.12
Embedded	Large, complex, strict constraints and time	3.6	1.2

So 2.4 and 1.05 are not arbitrary; they were fitted from historical software project data in Boehm’s study, specifically for small, simple “organic” projects.

COCOMO (Constructive Cost Model) Estimation

- COCOMO formula (Organic Mode):
- Assuming **20 working** days/month:
- Developer-days $= 41.8 \times 20 \approx 836$
- Cost $= 836 \times \$400 \approx \mathbf{\$334{,}400}$

- **Interpretation:** More systematic, accounts for size non-linearly.

Sources of Estimation Error

- Requirements volatility
- Lack of historical data
- Human bias
- Complexity underestimation
- Environmental factors

Factors Influencing Cost & Schedule Estimation

- Team experience
- Technology maturity
- Project size & complexity
- Development model (Agile vs Waterfall)
- Organizational culture

Risk Evaluation

Risk in Software Projects

- Definition: risk refers to any uncertain event or condition that, if it occurs, can positively or negatively affect project objectives such as time, and cost.
- Types: technical, managerial, financial, operational.

Risk Identification Frameworks

- Boehm's top 10 risk categories
- Risk breakdown structure (RBS)
- Risk in Global Software Development (GSD): timezone differences, culture, communication gaps.

Boehm’s top 10 risk categories

1. Personnel Shortfalls: Not enough skilled staff
2. Unrealistic Schedules and Budgets
3. Developers’ Inexperience with Project Application
4. Requirement Changes / Instability
5. Shortfalls in Early Planning / Specification
6. Customer Conflicts / Lack of User Involvement
7. Software Complexity: too complex that hard to test
8. Unrealistic Performance Expectations
9. Inadequate Tool Support / Technology
10. Organizational / Management Issues

Risk breakdown structure (RBS)

- **Technical Risks**
  - Inadequate testing
  - Technology failures
  - Performance bottlenecks
- **Project Management Risks**
  - Schedule slippage
  - Budget overruns
  - Poor planning/ requirement analysis
- **Organizational / External Risks**
  - Team turnover
  - Low stakeholder engagement
  - Resource unavailability
- **External / Environmental Risks:**
  - regulatory changes
  - market changes
  - natural disasters

Example RBS for Payroll Software Project

Risk: Level 1	**Risk: Level 2**	**Risk: Level 3 (Specific Risks)**
Technical	Complexity	Payroll calculation logic errors
Technical	Integration	Reporting module fails to integrate with HR database
Project Mgmt	Scheduling	M1 module delayed due to developer unavailability
Project Mgmt	Budget	Extra testing exceeds budget
Organizational	Personnel	Developer leaves project
External	Regulatory	Change in tax law or labor regulations

Risk vs. Uncertainty

- Risk: A situation where both the probability and impact of an event are known or can be estimated.
  - Example: You know that there's a 30% chance a third-party library will fail to integrate, causing $5,000 extra cost.
- Uncertainty: A situation where either the probability or impact is unknown or hard to predict.
  - Example: A sudden change in government regulations affects your software requirements. You can't assign a probability or exact impact.

Risk Response Planning

Response Strategy	**Description**	**Example**
Avoid	Change plan to eliminate risk	Switch to a proven technology
Mitigate	Reduce likelihood or impact	Add extra testing or training
Transfer	Shift risk to a third party	Buy insurance or outsource
Accept	Acknowledge risk, do nothing	Minor delays that can be tolerated

Risk Analysis & Assessment

- Probability vs impact
- Qualitative analysis (risk matrix)
- Quantitative analysis (expected monetary value)

Modeling Risk & Uncertainty

- Probability-Based Models
  - Use numerical probabilities to calculate Expected Monetary Value (EMV) or risk exposure.
  - Example: If a server outage has a 20% probability and $10,000 impact, EMV = 0.2 × 10,000 = $2,000.
- Scenario-based analysis
  - Consider different "what-if" scenarios for uncertain events.
  - A project could finish in 6 months (best) or 12 months (worst)
- Monte Carlo Simulation
  - runs thousands of project scenarios using probability distributions to predict project completion time or cost.
- Decision Trees
  - Visual representation of decisions, risks, probabilities, and outcomes.
  - Helps compare alternative strategies under uncertainty.

Monte Carlo Simulation

Suppose we are managing a software project with three main tasks:

Task	Minimum Time (days)	Most Likely Time (days)	Maximum Time (days)
Design	4	6	10
Coding	8	12	16
Testing	3	5	9

Objective: Estimate the total project duration considering uncertainty using Monte Carlo simulation. Also, determine the probability of completing within 22 days.

Step 1: Define Distributions

We use a triangular distribution for each task because we know the min, most likely, and max times.

- Design: min = 4, most likely = 6, max = 10
- Coding: min = 8, most likely = 12, max = 16
- Testing: min = 3, most likely = 5, max = 9

Step 2: Random Sampling (Simulation)

- We will simulate 5 iterations for simplicity (in real life, you'd run 10,000 iterations).

- Formula for triangular random sampling (simplified):
  Pick a random number between 0 and 1 → map to triangular distribution using approximate formula.

Step 2: Random Sampling (Simulation)

Iteration	Design (D)	Coding (C)	Testing (T)	Total Duration (D+C+T)
1	5	11	4	20
2	7	12	6	25
3	4	10	5	19
4	6	14	7	27
5	6	12	5	23

Step 3: Analyze Simulation Results

Iteration	Design (D)	Coding (C)	Testing (T)	Total Duration (D+C+T)
1	5	11	4	20
2	7	12	6	25
3	4	10	5	19
4	6	14	7	27
5	6	12	5	23

- **Average Project Duration:**

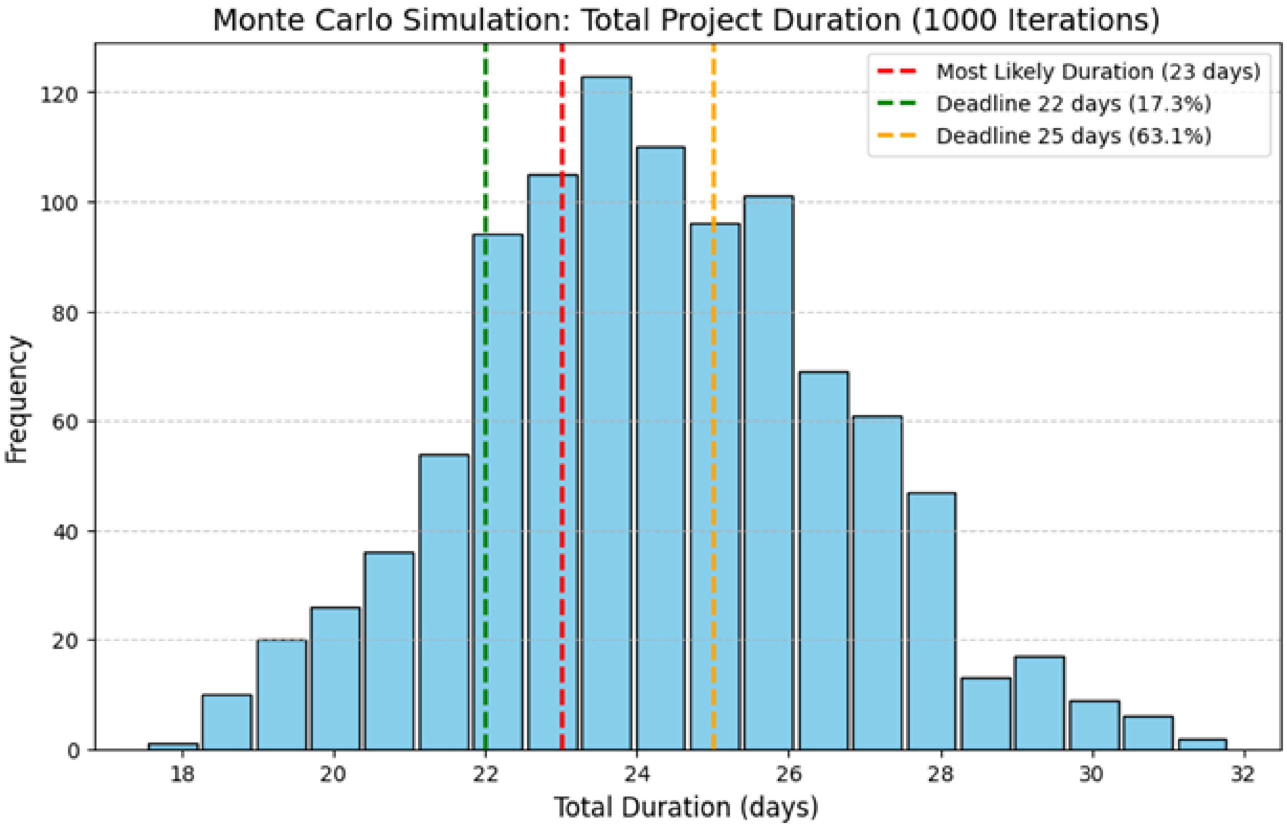
= (20+25+19+27+23)/5 = 114/5 = 22.8 days

- **Probability of finishing ≤ 22 days:**
  - Count iterations ≤ 22
  - Iteration 1 and 3 → 2 out of 5 → 40% probability.
- **Probability of finishing ≤ 25 days:**
  - Iterations 1, 2, 3, 5 →
  - 4 out of 5 → 80% probability.

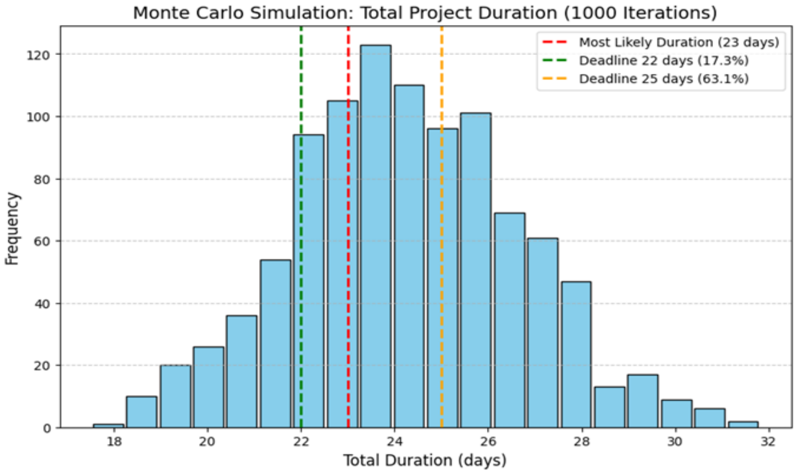
Step 4: Decision-Making

- If **deadline = 22 days**, the chance of success = 40% → High risk of delay.
- If **deadline = 25 days**, the chance of success = 80% → Acceptable risk.

Full-scale Monte Carlo simulation with 1,000 iterations



Monte Carlo simulation



- **Histogram** shows the distribution of total project durat
- The **red dashed line** marks the most likely duration (~23 days).
- The **green dashed line** marks a 22-day deadline: probability of finishing on time = **~17.3%**.
- The **orange dashed line** marks a 25-day deadline: probability of finishing on time = **~63.1%**.

End of Criteria for Project Evaluation