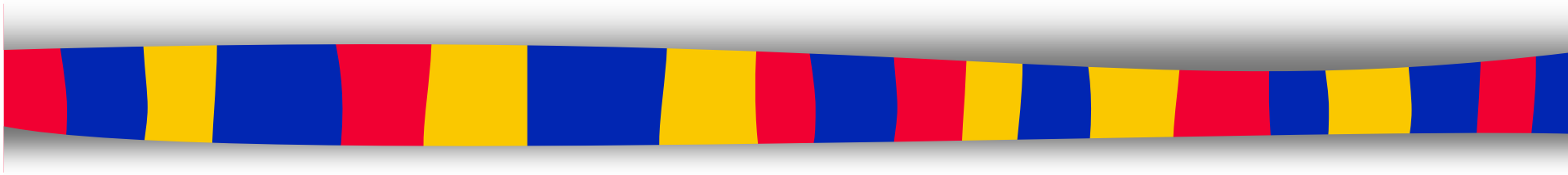
# CISC 322

## Software Architecture



## Lecture 19:

## Software Cost Estimation

**Emad Shihab**

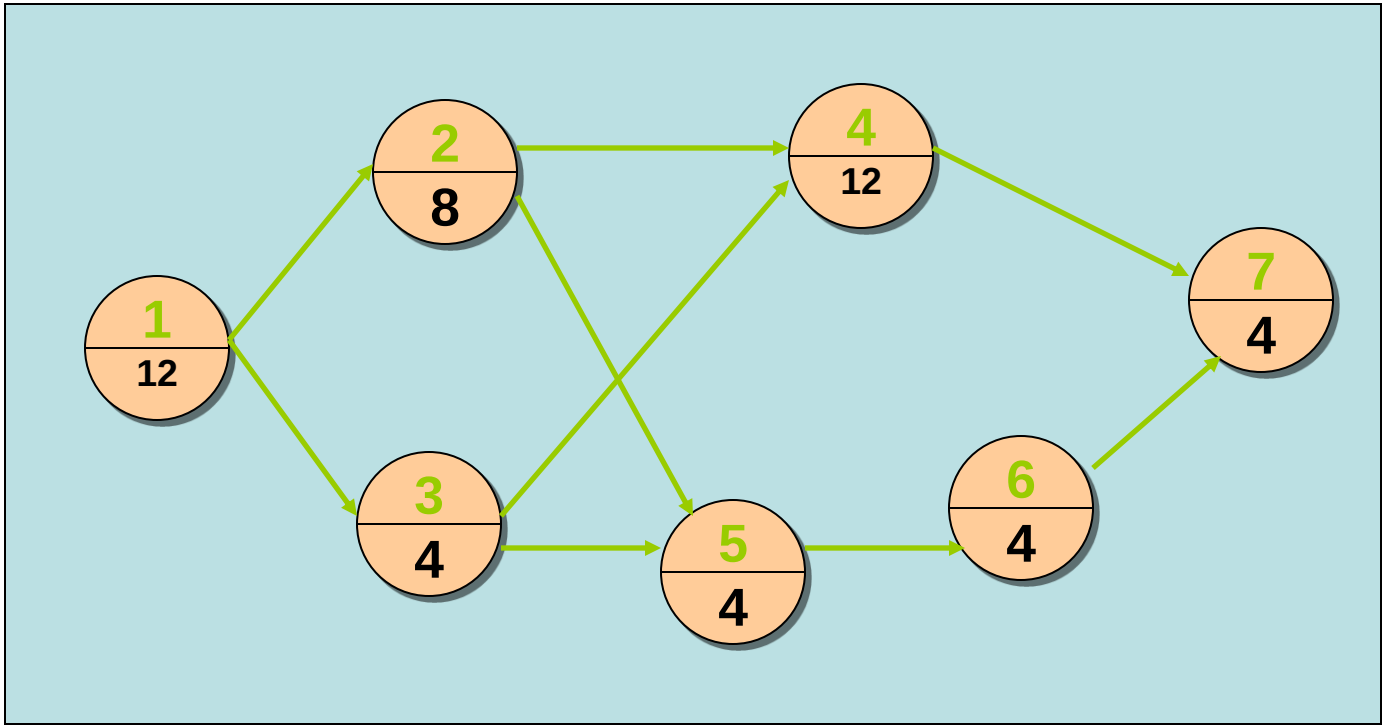
**Slides adapted from Ian Sommerville**

# Last Class

- Program Evaluation and Review Technique (PERT)
  - Determine critical path
  - Calculate prob. that a project finishes within X weeks
- Project crashing

# Cost Estimation

# Why Cost Estimation?



# Why Cost Estimation?

- Need to establish a budget
- Need to set a price
- Need to make a profit

# Cost Estimation

- Cost estimation and scheduling are usually done together
- Cost is driven by three main activities:
  - HW and SW costs, including maintenance
  - Travel and training (can be reduced using technology)
  - Effort costs (paying personnel)
- For most projects **effort costs** is the dominant cost

# Effort Costs

- Effort costs are more than just salaries
  - E.g., heat, lighting, support staff, networking, recreational facilities, security, etc…
- Effort cost is calculated by taking the total cost of running the organization and dividing by number of productive staff
- How much does overhead cost?

# Cost Estimation Topics

- Productivity
- Estimation Techniques
- Algorithmic Cost Estimation
- Project Duration Staffing

# Software Productivity

- Generally, productivity is measured as:
  - Number of units/ person hours
- Not the case in software…why?
- Can have many solutions
  - Solution 1: executes more efficiently
  - Solution 2: easier to read and maintain

# Software Productivity

- Based on measuring attributes of the software divided by total development effort
- Size related: LOC delivered
- Function related: Function points and object points

# Size related metrics

- LOC per programmer-month (LOC/pm)
- This time includes requirements, design, coding, testing, documentation
- Advantage: Easy to calculate
- Disadvantage: different languages
  - E.g., 5000 assembly ~ 1500 C

# Function Related Metrics

- Productivity = FP/pm
- FP is related to:
  - External and internal inputs
  - User interactions
  - External interfaces
  - Files used by the system
- Functionality is independent of implementation language

# Function Points

- Some input and output interactions, etc. are more complex than others
- You can give a weight to the FP, considering:
  - Amount of reuse, performance, etc…
- FP count is highly subjective and depends on the estimator!
- FPs are biased towards data-processing systems

# Object Points

- Are an alternative to FPs
- The number of object points is a weighted estimate of:
  - No. of separate screens displayed (1,2,3)
  - No. of reports produced (2,5,8)
  - No. of modules that must be developed to support 4$^{th}$ generation language code

# FP and OP

- OP are easier to estimate. They only consider screens, reports and modules
- OP can be estimated early in the development process
- Can approximate LOC from FP or OP:
  - LOC = AVC x No. of FP
- AVC is 200-300 LOC/FP in assembly language and 2-40 LOC in 4GL

# Productivity Estimates

- Many factors impact productivity
  - Some programmers are 10 times more productive
  - Application domain:
    - Embedded systems: ~30 LOC/pm
    - Application systems: ~900 LOC/pm
    - 4-50 OP/pm, depending on application, tools, developers
  - Process, project size, technology support, working environment

# LOC don’t impress me much!

- Counting LOC does not take into account:
  - Reused code
  - Generated code
  - Quality
  - Performance
  - Maintainability
- Not clear how productivity and quality metrics are related!

# Estimation Techniques

- There is no simple way to make accurate estimates of the effort required
  - Initially, not much detail is given
  - Technologies and people may be unknown
- Project cost estimates may be self-fulfilling
  - Estimate defines budget, project adjusted to meet budget

# Many Estimation Techniques

- Algorithmic cost modeling
- Expert judgment
- Estimation by analogy
- Parkinson’s Law
- Pricing to win

# Algorithmic code modelling

- Model is built based on historical cost information
- Generally based on the size of the software

# Expert judgement

- Several experts in software development and the application domain are consulted
- Process iterates until some consensus is reached
- Advantages: Relatively cheap estimation method. Can be accurate if experts have direct experience of similar systems
- Disadvantages: Very inaccurate if there are no experts!

# Estimation by analogy

- The project is compared to a similar project in the same application domain
- Advantages:  Accurate if project data available
- Disadvantages: Impossible if no comparable project has been tackled

# Parkinson's Law

- “Work expands to fill the time available” i.e., the project costs whatever resources are available
- Advantages:  No overspending
- Disadvantages: System is usually unfinished

# Pricing to win

- The project costs whatever the customer has to spend on it
- Advantages: You get the contract
- Disadvantages: The probability that the customer gets the system he or she wants is small. Often, costs do not accurately reflect the work required

# Cost Estimation Approaches

- The aforementioned techniques may be used top-down or bottom-up
- **Top-down**: Starts at the system level and assess system functionality and its delivery through subsystems
- **Bottom-up**: Start at component level and aggregate to obtain system effort

# Top-down vs. Bottom-up

- Top-down:
  - Usable without much knowledge
  - Factors in integration, configuration and documentation costs
  - Can underestimate low-level problems
- Bottom-up:
  - Usable when architecture of the system is known
  - May underestimate system-level activities such as integration

# Algorithmic Cost Modeling

- A cost model can be built by analyzing the cost and attributes of similar projects

- **Effort** = A x $Size^{B}$ x M

- **A** – depends on organization
- **B** – ~1-1.5 reflects disproportionate effort for large projects (comm. and conf. management)
- **M** – reflects product, process and people attributes

# Estimation Accuracy

- Difficult to estimate size early on. B and M are subjective
- Several factors influence the final size
  - Use of COTS and components
  - Programming language
- Estimations become more accurate as development progresses

[Sommerville 2000]

# Estimate uncertainty