Software Estimation

What you need to know

Mel Rosso-Llopart
rossollo@cmu.edu

Session Objectives

- Where is the software industry with estimation of project
  - Why are we bad at estimating?
- What do we mean by a ROM (Rough order of magnitude)
- What are some different ways of quickly estimating an effort
  - Uses of cost estimates
- What are the Critical factors we must consider when estimating software projects

My Background

- B.S., Physics and Biology, University of California, Riverside
- BS, Computer Science, University of California, Riverside
- 4/1983-11/1998, Department of Defense, Edwards AFB California
- Graduate MSE program 1992
- Software Engineering manager, Adtranz Train Division
- 2000-present, Assoc. Professor, Carnegie Mellon

Why Estimate Software?

- 30% of project never complete
- 100-200% cost overruns not uncommon
- Average project exceeds cost by 90%; Schedule by 120%
- 15% of large project never deliver anything
- Only 16.2% of projects are successful

* 1998, 1999, 2000 Standish report, Choas

Problem for Estimates

- Parkinson's Law
  - Cost will rise to meet budget
  - Work expands to fill the resources available

What are the consequences?

- Economic
  - Loss of contracts
  - Company failure
- Technical
  - Dependency on software growing
- Managerial
  - Change of personnel

Why are we bad at estimating?

- Complexity of the systems
  - Infrequency - How often do we do the "same thing"
    - Vs manufacturing or construction
  - Underestimation bias
    - Computers are "easy"; software is "easy"
  - We deal with Goals not estimates
    - Must be done by June
  - Complexity is what makes estimating hard

Why are we bad at estimating? (2)

- Complexity of the systems
  - ~1000 FP in a pace maker (50K)
  - ~18,800 FP in shuttle test scaffolding (1,000,000 LOC)
  - ~75,400 FP in Nynex Switch (4,000,000LOC)
- "Human brain capacity is more or less fixed, but software complexity grows at least as fast as the square of the size of the project"

Tony Bowden

Early Estimation
In the bid for example

- No "real" money in the bid
- Must estimate on your dollar
- What is important for this estimate
  - Can I compare to history
- Done as quickly and cheaply as possible
- How important is it?

Early Size Estimation Critical, Yet Dangerous

- You need to know size in order to bid, but the time you know the least about the real size is at the beginning
- Construction analogy: given some set of metrics [square footage, number of corners, volume of concrete, etc] a contractor can estimate cost; does this relate to software?

Estimation techniques

- No simple way to make accurate estimates of the effort for a software system
  - Initial estimates based on inadequate information
    - user requirements definition
  - Software may run on unfamiliar environments
    - Different computers or new technology
  - The people in the project may be unknown
- Project cost estimates may be self-fulfilling
  - The estimate defines the budget and the product is adjusted to meet the budget

Computing Project costs

- Development effort is known
  - Person-Months
- Cost per development unit is known

Cost of Project =

Number of Person-Months X

Weighted average cost* per month

* Weighted average cost = burdened cost and can be 2 to 3 times salary.

Problem

- The accuracy of the previous equation depends on what?
  - Project Cost = Time X Unit Cost
- Accuracy of the development effort estimate
- Accuracy of the cost per unit
- Which one do we normally know?

Determining "development effort"

- Development effort measures
  - Person-Month
  - LOC per Hour
  - Function point per hour
  - Requirement per hour
- Most common is person-months (or hours)
- Let's look at ways to get development effort

First look for "similarities"

- Have we done something similar
- Do we have data on that project
- How long ago was it
  - Geometric loss of understanding
- Do we still have the expertise
  - Expertise does not last
- Do we have the artifacts from that project
  - Can we read them

Next look for "differences"

- Do we understand the differences
- Do we have expertise in this new area
  - Training cost time and money
  - Can we get the expertise quickly
- Do we have a proxy for this difference
  - Have we done something similar on other projects

Conceptual design

- Can we create a rough solution
  - End to end
  - How big or small should the parts be
- Can we estimate the parts
- Never confuse Conceptual with actual design
  - This is for estimating, you will redo if you win the bid

Conceptual Design

- Never, *ever*, confuse this with the "real" design
- Objective is to make this as fine-grained as possible without making commitments

Estimating exercise 1

- How many tennis balls will it take to fill this room?
  - How would you go about making the estimate?
  - What do you need to know?
  - What assumptions would you make?

Estimating exercise 2

- If the project is well understood
  - 2 months to deliver (40 days)
  - 25 LOC per day per engineer
  - Estimated 5000 LOC
  - How many people needed?

What are the major assumptions above?

Other methods: Large Scale Analogy

- Similar size of projects
- Similar type of projects
- Similar size of components
- Similar type of components
- Requirements type and size...
- Create a handbook of estimating history you can use
- No data makes it harder

Improving Your Chances: ***Wideband Delphi***

Six step process

- Planning – define the scope of the problem
  - Break large problems into smaller
- The Kickoff – To deliver problem to team
- Individual preparation – Everyone does individual estimates on problem parts
  - All assumptions are written down
- Estimation Meeting – Everyone on team gets together
- Assembling Tasks – Put together the whole project of estimates
- Review Results – Bring team back to review final results

The Delphi process in Wideband

- Estimation Meeting
  - A moderator collects the estimates for the task being estimated
    - Present the average or a line with all estimates (anonymous)
  - The estimate is discussed and assumptions presented
  - Moderator calls for a new estimate from everyone
  - Values are again presented to the team as average or line
  - Continue process until:
    - Four rounds are completed
    - The estimates "converged" to an acceptably narrow range
    - The allotted meeting time is complete
    - All participants are unwilling to change their estimates
  - 15-20 minutes per item discussed

Rounds in Delphi

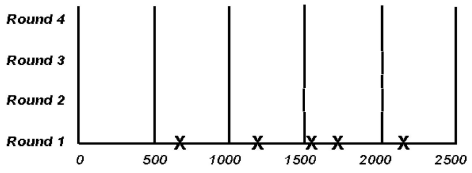


Figure 4. Estimation chart showing three rounds from a Wideband Delphi session.

Rules to insure best results for Wideband Delphi

- Gather a heterogeneous team
- Write down assumptions
- Make anonymous estimates
- Never estimate tasks larger than you are comfortable with
- This is "estimation" not "prediction"

Clark's Method

- Modularize the product
- Make best estimate per module
- Determine upper and lower size limits
- Size = $\sum$ module sizes which =

$$\frac{\text{Largest} + 4 * \text{Middle} + \text{Smallest}}{6}$$

Uses of Cost estimation

- Project planning and Control
  - Best and timely use of resources
- Budgeting
  - Planning and allocation of resources
  - Bidding
- Software design improvement
  - Find a first solution (Is there a better one?)
- Risk Analysis

Estimating critical factors

- X 5 for Safety critical software (ASWEC'97)
- X 5 for embedded systems (S/W Eng, 6th Ed.; Ian Sommerville)
- X 1.5 for Reusable code
- X 2.5 for distributed (Herbsleb, IEEE June 2003)

Outsourcing issues (higher cost)*

- Additional communication
  - Interval between questions and answers
- Distributed social networks not as affective as local networks
- Modification requests take longer to coordinate

* J.D.Herbsleb and A.Mockus "An Empirical Study…Software development, IEEE June 2003

Past experience

- Experience is the "worst teacher"
- …it gives the test first and the lesson after
- Estimating gets better the more you do it
- Need history to get better at estimating

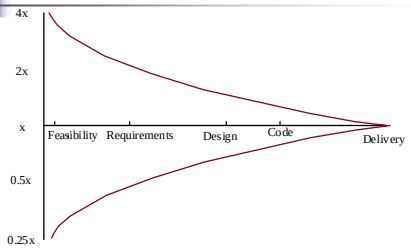
Experience-based estimates

- Estimating is primarily experience-based
- However, new methods and technologies may make estimating based on experience inaccurate
  - Object oriented rather than function-oriented development
  - Client-server systems rather than mainframe systems
  - Off the shelf components
  - Component-based software engineering
  - CASE tools and program generators

Estimation accuracy

- The size of a software system can only be known accurately when it is finished
- Several factors influence the final size
  - Use of COTS and components
  - Programming language
  - Distribution of system
- As the development process progresses then the size estimate becomes more accurate

Estimate uncertainty



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. Costs do not accurately reflect the work required

Summary

- You know now why we are bad at estimating
- You should understand what it means to create a ROM
- Some quick methods of estimating
  - Clark Method, Wideband Delphi, FP, etc.
- You know some of the Critical factors involved with estimating software projects

Questions? Comments?