

The Limitations of Estimation

Linda M. Laird

Linda_m_laird@msn.com

June, 2006



“Anyone who expects a quick and easy solution to the multi-faceted problem of resource estimation is going to be disappointed.” 1968, Alfred M. Pietrasanta at IBM System Research Institute

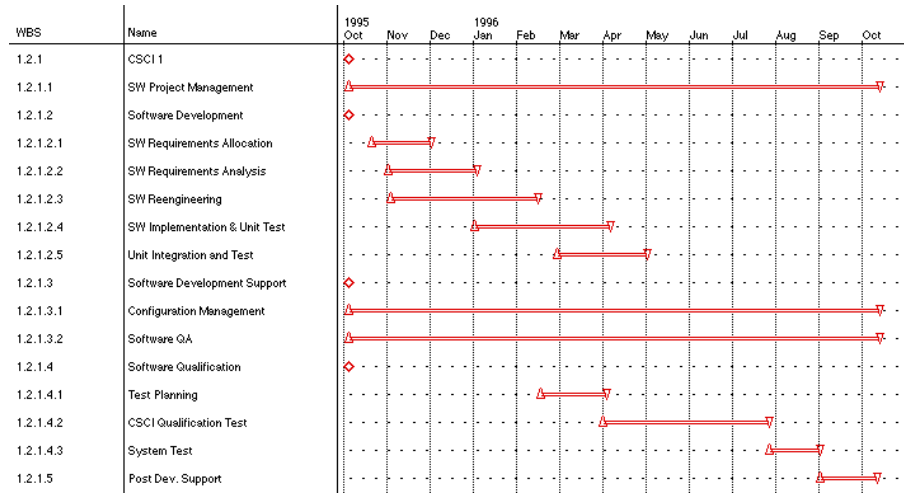


Agenda

- Why Estimate?
- Estimation Exercise
- Estimation Model
- Software Estimation – Engineering or Art
- Handling Uncertainties
- Combining Estimates
- Final Recommendations



What are we trying to estimate?



How do we do that?

- Most models & methodologies start by estimating Size (KLOC, FPs, Use Case Points) first....and using that to drive to Effort (e.g., staff days/months/years)
- And then use Effort to determine
 - Staffing
 - Schedule
 - \$\$
- Determining Effort is the big step
 - Staffing, schedule, and \$\$ all come from effort



How do you currently estimate effort for a software project?



□ How well do you do estimate?

- Typically, if you under estimate, you miss schedules. If you over estimate, then the work expands to meet the estimate....



○ Do you know?



□ How is the industry doing?



Some estimation data

- Standish Group's Chaos Report
 - 89% overruns
- Other Surveys
 - 30-40% over runs - most common value reported

Ref: "A Review of Surveys on Software Effort Estimation"
: Molokken and Jorgensen, Simula Research Laboratory,
2003



Reasons for failure

□ Mostly commonly stated

- Specifying incomplete or unclear requirements
- Failing to adjust schedules when scope changes
- Setting development schedules that are too aggressive
- Insufficient resources.

□ Note: the last three really say

- Too much work
- Too little time
- Not enough people/equipment.

□ Reasons for Overruns are complex

- “May have tendency to over-report causes that lie outside of their responsibility...e.g., customer-related causes”

Ref: “A Review of Surveys on Software Effort Estimation”
: Molokken and Jorgensen, Simula Research Laboratory,
2003



Root Causes – My experiences

- ❑ Many don't know how to estimate well, nor practice (with any kind of feedback)
- ❑ Confusion of target/goal with estimate – and being pushed into a date because it is what the business needs
- ❑ Hope (which leads to over-optimistic planning)
- ❑ Inability of Software Engineers to credibly communicate and support their estimates
 - Lack of objective/quantitative understanding of estimation tools and techniques
- ❑ Creeping features
- ❑ Quality “Surprises”



My bias.....Our Software Education

- Most of us are taught computer science.....
- Few are taught software engineering.....



Do we care if we can estimate well?



Yes --
Because
our
companies
care.....



□ Yes – Because
it is the only
way to have a
personal life if
you want to
stay in this
business



What is the value of better estimation?

- ❑ Improves the quality of Decision Making
 - Allows better choices to be made
- ❑ Gives us the opportunity to adjust project parameters to meet cost/schedule budgets/deadlines
 - E.G., By understanding the cost factors, you can adjust them to come in closer to the target.
- ❑ Gives us the opportunity to manage risk, once risk is known



□ In summary ...we need to and want to do a better job of estimation...



□ Estimation Exercise!



How long will it take to walk from this room to the Leib Building?



□ How long will it take to walk from this room to the Leib Building?

- The Computer Science Department is in Leib
- It is .75 miles



How long will it take to walk from this room to the Leib Building?

- The Computer Science Department is in Leib
- It is .75 miles - as the crow flies...and there is construction so you have to take a detour which will make it 1/3 longer



How long will it take to walk from this room to the Leib Building?

- The Computer Science Department is in Leib
- It is .75 miles - as the crow flies...and there is construction so you have to take a detour which will make it 1/3 longer
- You are not the one who is walking – it is a 19 year old student who is late for class



How long will it take to walk from this room to the Leib Building?

- The Computer Science Department is in Leib
- It is .75 miles - as the crow flies...and there is construction so you have to take a detour which will make it 1/3 longer
- You are not the one who is walking – it is a 19 year old student who is late for class - who can walk/run a mile in 11 minutes



How long will it take to walk from this room to the Leib Building?

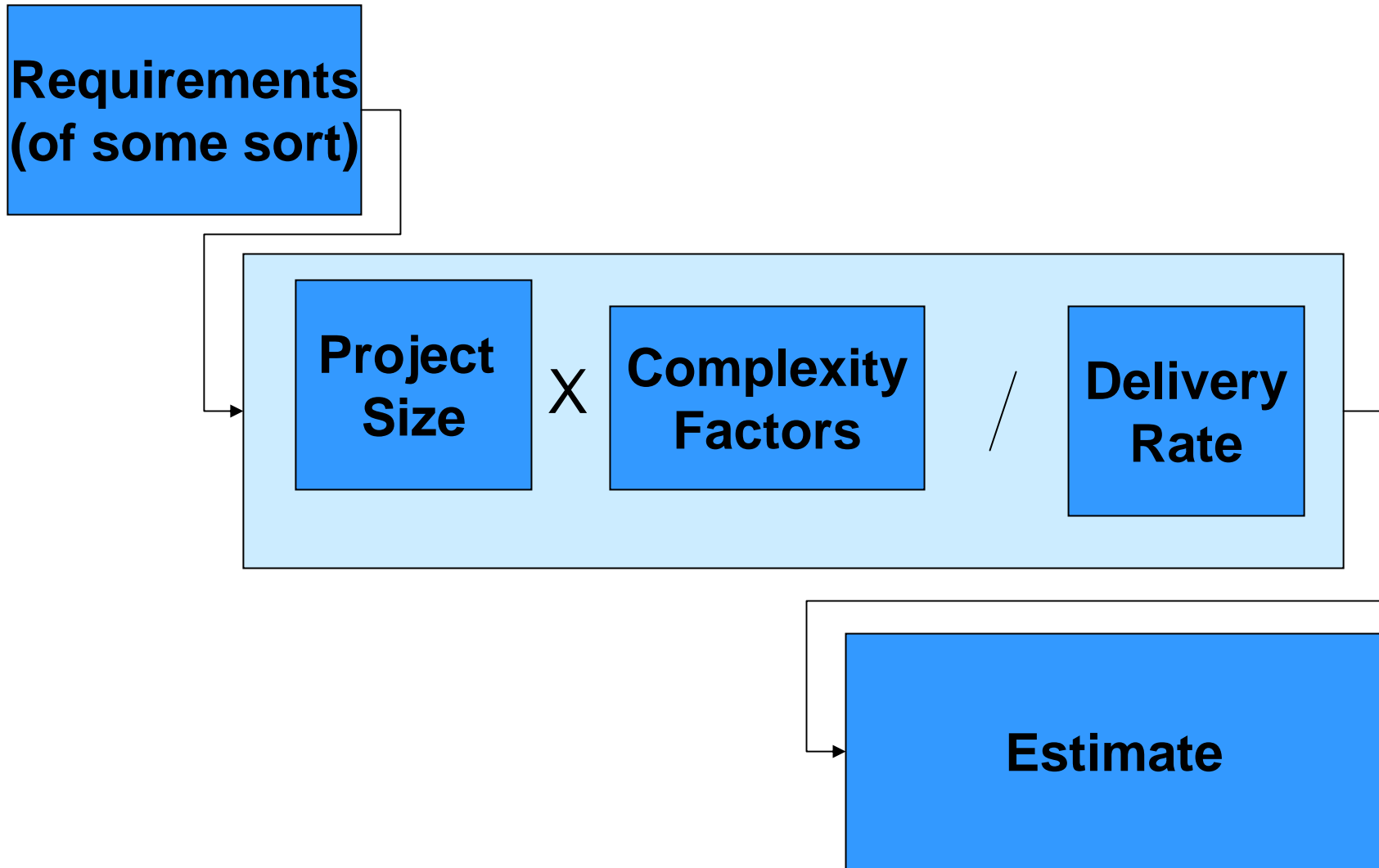
- The Computer Science Department is in Leib
- It is .75 miles - as the crow flies...and there is construction so you have to take a detour which will make it 1/3 longer
- You are not the one who is walking – it is a 19 year old student who is late for class who can walk/run a mile in 11 minutes
- The elevators don't work



- What did we just do?
- How is it similar to software development effort estimation?



Estimation Model



□ Estimation Methods and Models



Software Estimating Methods & Models

- Over ~~40~~ 100 Methods & Models Documented
 - Some old ones based SW cost on % of HW cost!
 - Output for most is a estimation of staff effort from a wide range of inputs
 - Most are a methodology that is an analytical formula that takes as parameters system characteristics, system size, complexity, and development environment—which are typically called “cost drivers”
 - Most only cover “software design, development & testing” -- need to make sure what is included
 - Significant limitations on all of the models

Source: Wellman, p.11



Model Approaches

□ Top Down

- Overall estimate based on the Global Properties
- Advantage: It follows the structure of the software engineering processes and the natural evolution of the design/development process
- Disadvantage: Difficult technical problems may be masked and cause later cost overruns

□ Bottom-up

- Involves costing of components in the form they will be developed...related to the work breakdown structure
- Disadvantage: Skill of estimators; Can not be done until well-defined design
- Complementary with Top-Down at Different Points in Process

Source: Wellman, p.11



Use of Major Models

- N.B. Some models work well frequently, some work well occasionally, all are occasionally horrible.
- Studies on models:
 - For a well-documented 9KLOC system:
Variation from 6.6KLOC to 37KLOC
 - For a hypothetical system – range from \$325K to \$2,750K – with staff unit costs being identical

Source: Fenton & Plegger and



My experiences with my students

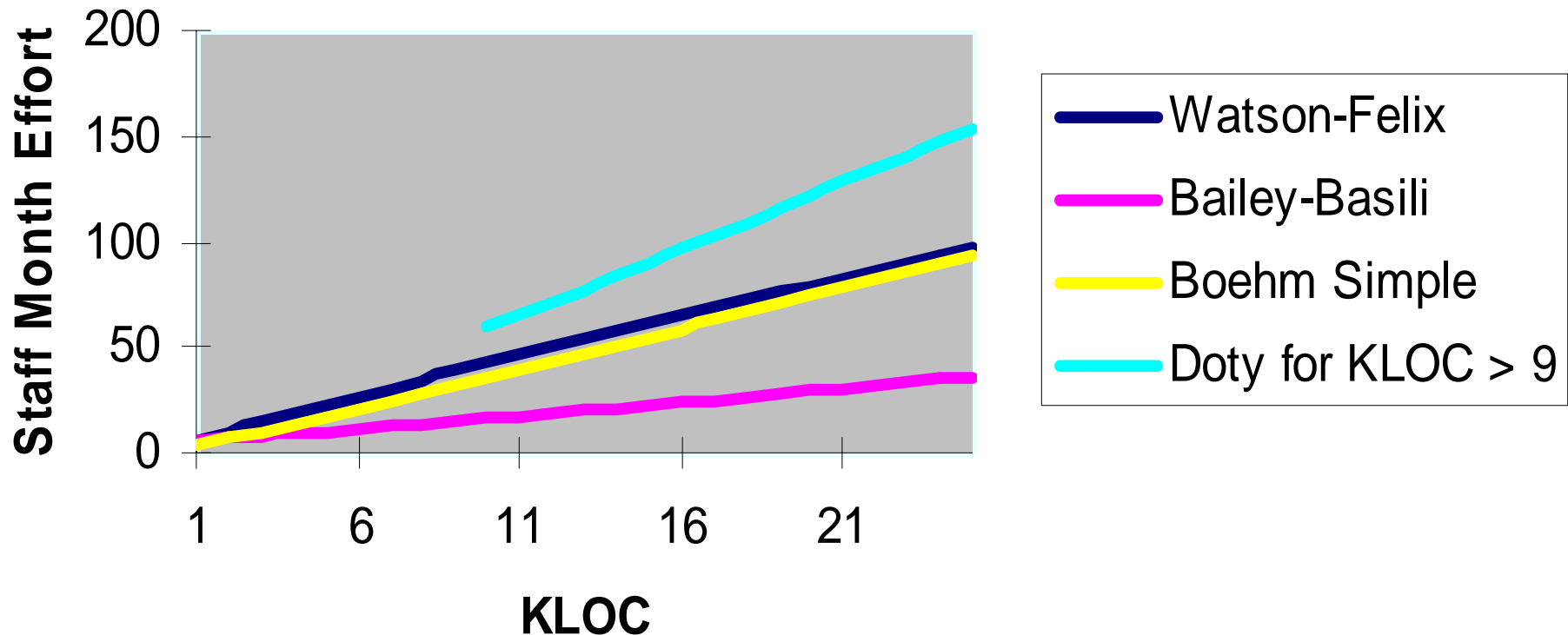
- I teach over 10 different ways to estimate effort – which can be categorized as
 - Expert opinion
 - Using Benchmark Data
 - Analogy
 - Proxy Points
 - Custom Models
 - Algorithmic Models
- The students estimate the same project using most of the different models
 - The individual estimates vary wildly
 - Due to different definitions of the projects, assumptions, as well as models.



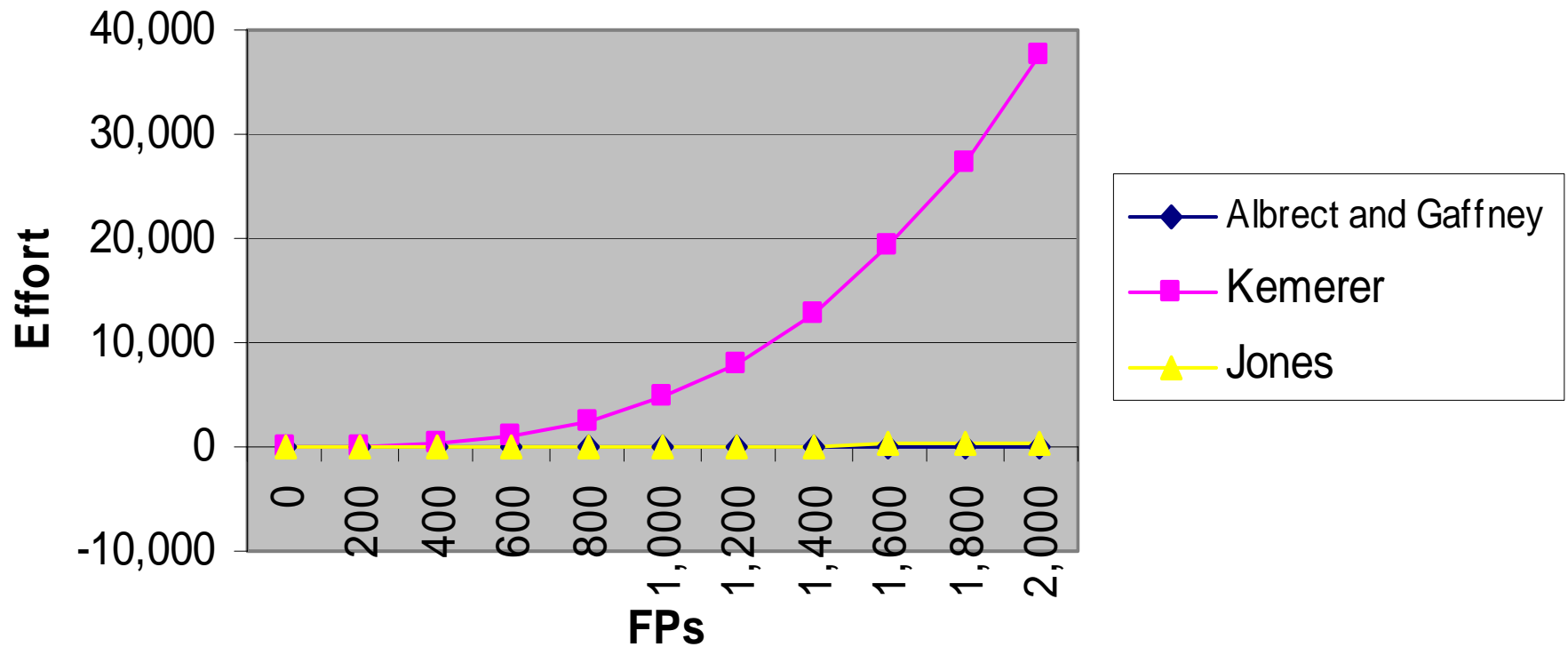
Let's look at some simple algorithms



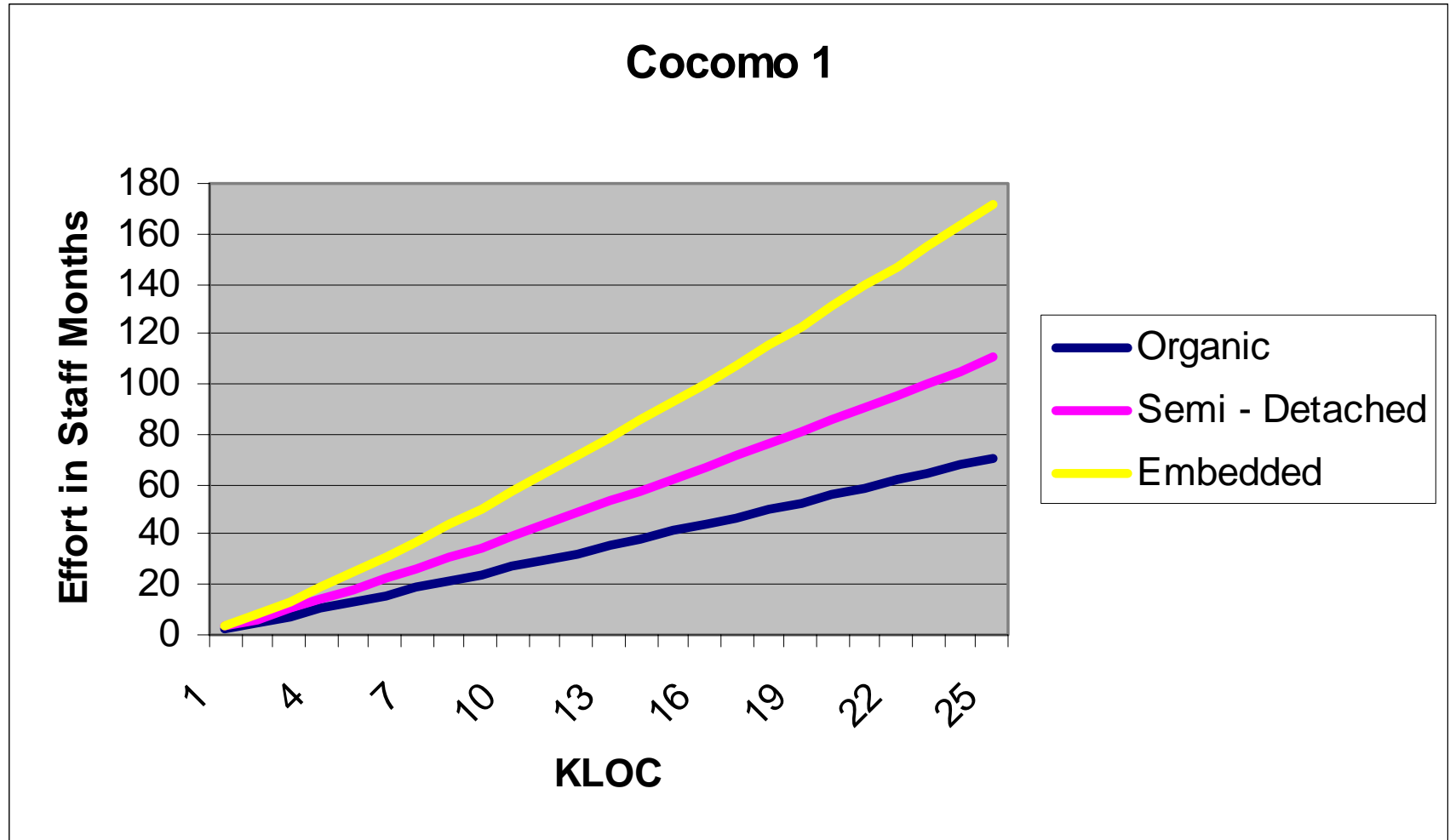
Comparison of Algorithmic KLOC Modles



Effort Predicted by FPs



Cocomo 1 Estimations



- 15 attributes for a product, and a six point scale
- Use the weighted attributes to adjust the basic equations.
- Boehm states that intermediate Cocomo produces sound results “within 20%, 68% of the time.” ... e.g., 1SD



What do you think of these results?

“Within 20%, 68% of the time”



Use Case Point Studies – Bente Anda – Simula Research Labs

Company	Project	Use Case Estimate	Expert Estimate	Actual Effort	Deviation use case est.	Deviation exp. est.
Mogul	A	2550	2730	3670	-31%	-26%
Mogul	B	2730	2340	2860	-5%	-18%
Mogul	C	2080	2100	2740	-24%	-23%
CGE&Y	A	10831	7000	10043	+8%	-30%
CGE&Y	B	14965	12600	12000	+25%	+5%
IBM	A	4086	2772*	3360	+22%	-18%
Students project	A	666		742	-10%	
Students project	B	487		396	+23%	
Students project	C	488		673	-25%	



□ Why aren't the results better?



Is software estimation an engineering process? Or art?.... Or impossible?



□ “...if software estimation is believed to be a codifiable engineering process analogous to house building then litigation is a reasonable and expected consequence of inaccurate estimations.”

Terry Bollinger, "The Interplay of Art and Science in Software," *Computer*, vol. 30, no. 10, pp. 128,125-127, Oct., 1997.



2 Camps in SE Community

- The “process” camp:
 - Quality software can be developed on time if a particular software process or programming technology is used
- The “problem solving” camp:
 - Programming is fundamentally a process of solving problems and as such intrinsically resists codification.



- “The creation of genuinely new software has far more in common with developing a new theory of physics than it does with producing cars or watches on an assembly line.” --- T. Bollinger



Software Development: Process Execution or Problem Solving?



Or



- What do you think? Or the execution of a process? Is programming problem solving?

- Does it vary, based upon the circumstance?



Limitations of Estimation – Why?

- The question is whether you are dealing with anything new or unknown

- ... You almost always are...

- The questions become how to bound and manage the uncertainties...

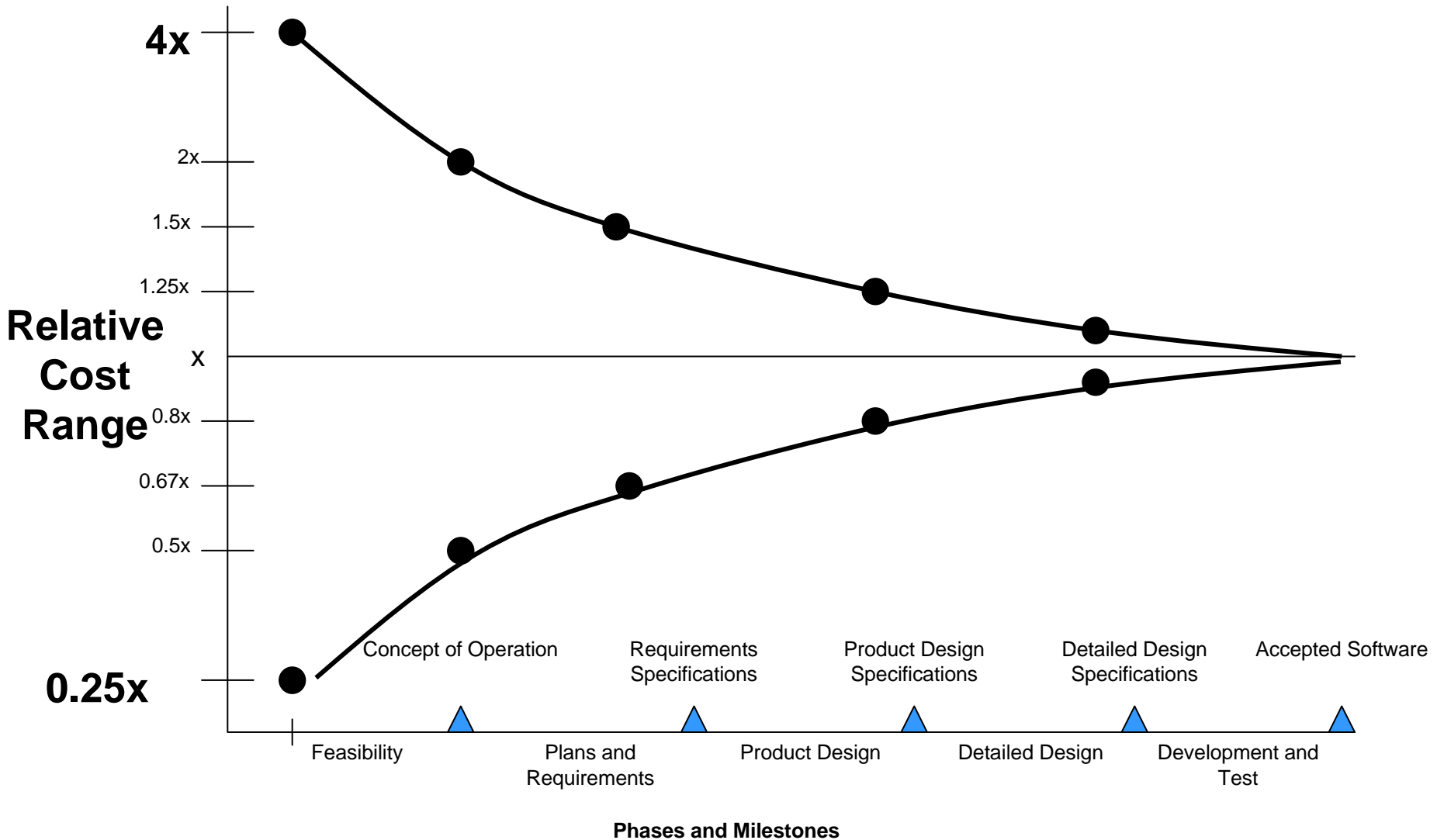


- Models and Methodologies are not a replacement for Common Sense and Engineering Judgment

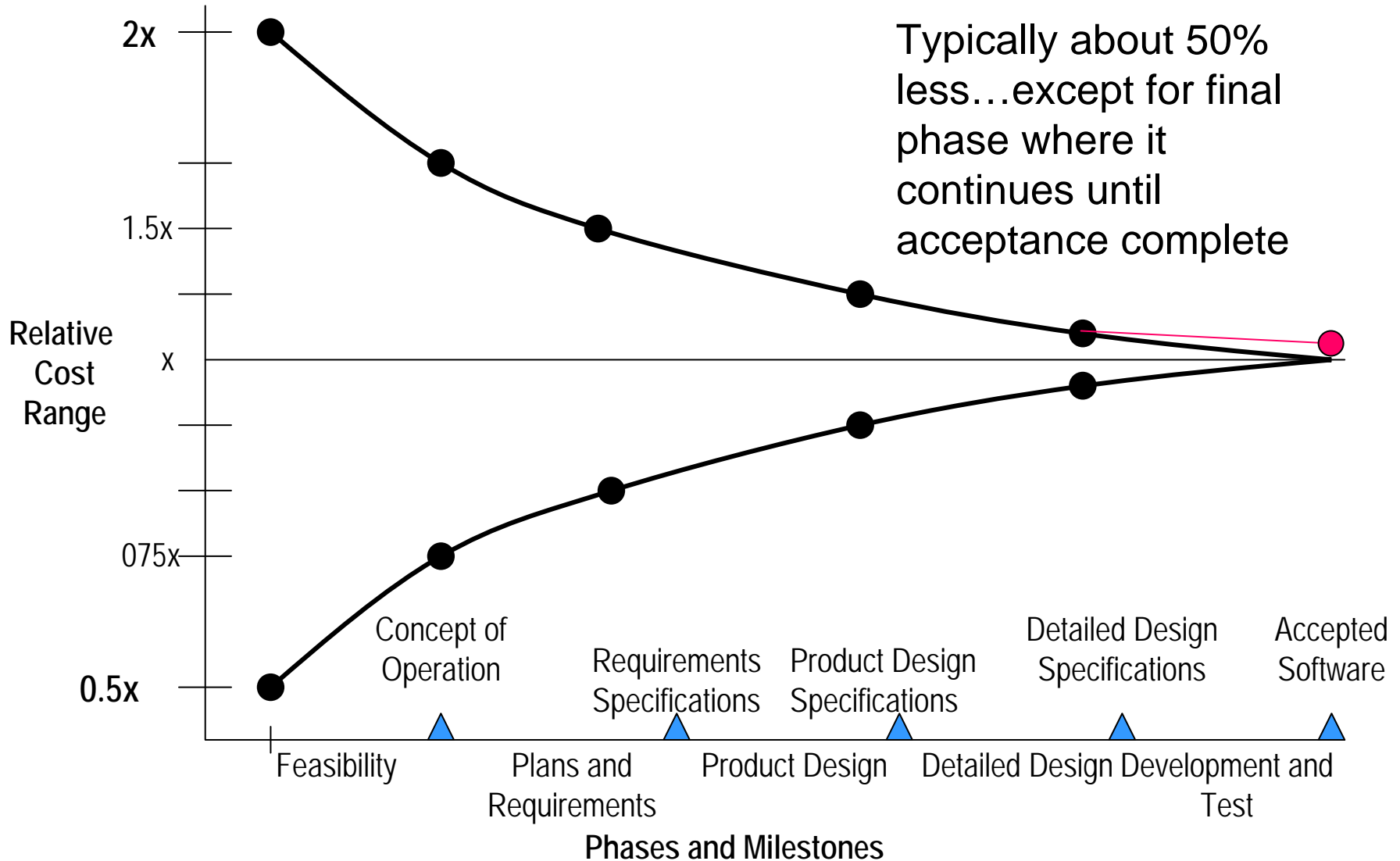
- Need to use all four of them



Barry Boehm's Cone of Uncertainty - 1981

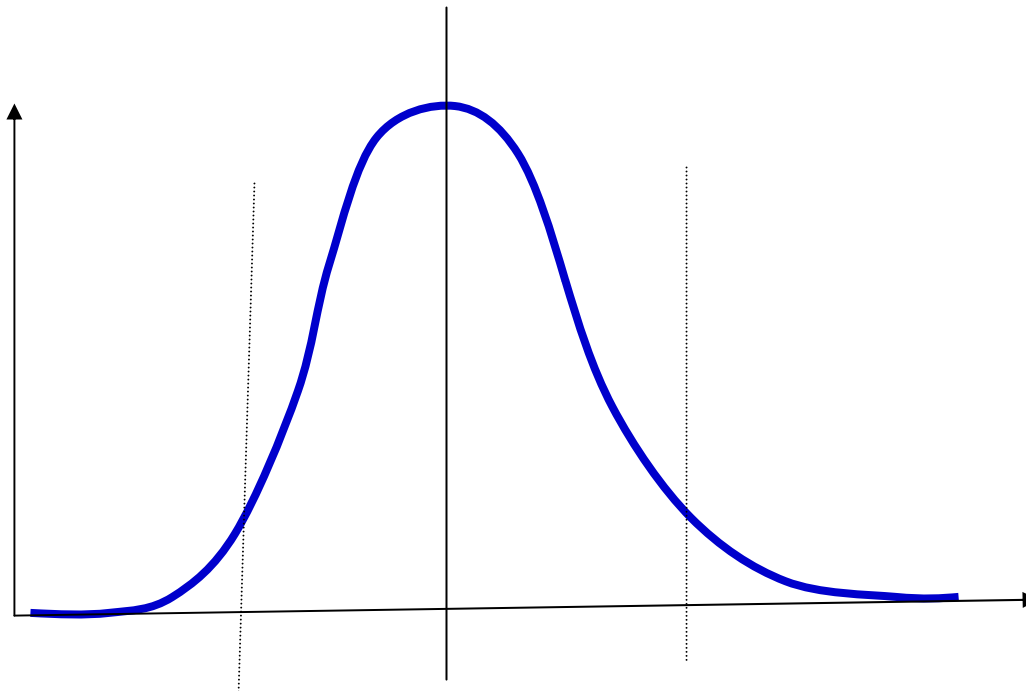


Today's View---Approximately



Remember what an estimate is...

- It tends to be the most likely view...and it is equally likely to be over as well as under.....remember statistics...

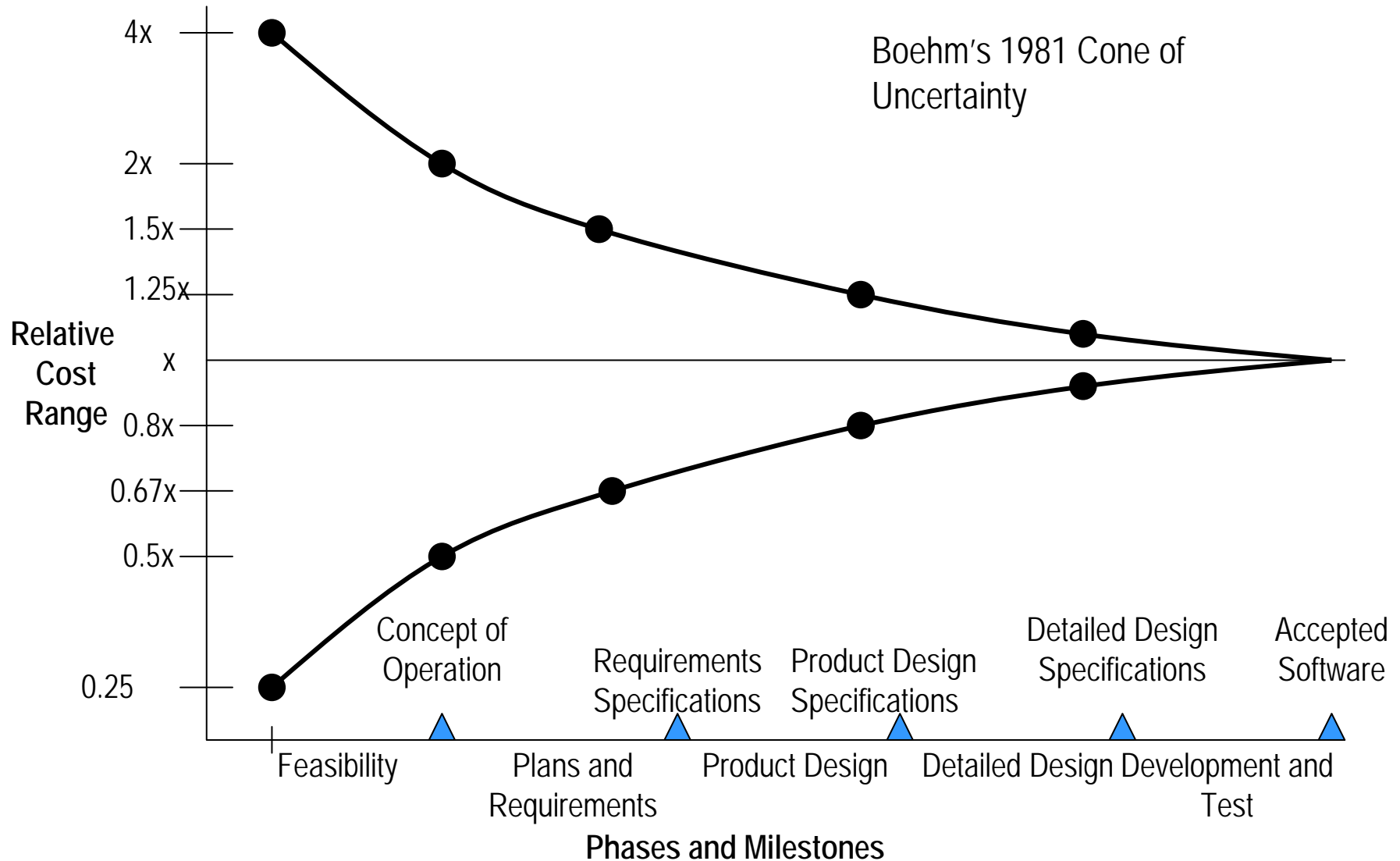


We need to

- Accept and consciously deal with Estimation Uncertainties
 - Understand, accept, and manage the inherent uncertainties in estimation
 - Change our terminology in estimation to include the uncertainties



Uncertainty varies over time



What are estimation uncertainties?

□ Statistical Variance

- Always will exist
- Only question is size
- Repeatable Processes -> Reduction in this variance

□ Known Risks

- Use Risk Management Plan
- Add contingencies

□ Unknown Risks

- Use experts to minimize them
- Use Management Contingency
 - Based on expected invention required
 - PMI recommends 10% if nothing else known



□ Which is better?

- 12 staff months
- 12 staff months – 50% of the time it will be less, 50% of the time, it will be greater (except for parkinson's law)
- 95% of the time between 10 and 14 staff months
- 95% of the time \leq 14 staff months



Estimation Terminology

- Estimations are probabilistic - Need to reflect probability and uncertainties in estimations
- X% View
 - Means there is a X% chance of not exceeding the estimate
- Probability Intervals -- Ranges -- representing different probabilities
 - Between Y and Z staff months or even
 - 90% view is that it will be between Y and Z staff months



Determining Probabilities

- ❑ 50% view is typically what is estimated
- ❑ Experts (and some tools) will give you the ranges as part of the estimation process
- ❑ Unfortunately, studies have shown that the experts are typically over-confident.
- ❑ More accurate ranges from organizational baseline engineering rules or default engineering rules



Example: From project data

- Assume your estimate is 10 staff years, and on previous similar projects,
 - 30% came in on budget,
 - 10% came in under budget by 10%,
 - 40% came in over budget by 50% and
 - 20 % came in over budget by > 50%.

- Then your %estimates are:
 - 10% estimate - 9 staff years,
 - 40% estimate - 10 staff years
 - 80% estimate – 15 staff years



Estimation Uncertainty using Engineering Rules

□ Defaults by Phase:

- Feasibility + 100%, - 50%
- Rqmts + 50%, - 25%
- Design + 20%, - 10%
- Coding +10%, - 5%
- Testing +5%, - 2.5%

□ Example:

- Your current 50% estimate is 30 staff months, and you are in the design phase
- You'd say that the expected range is between 36 and 27 staff months



Estimation Uncertainties - Summary

- Change your terminology to X% views and ranges
- Use methods other than gut feelings for determining the ranges



Combining Estimates

- Always recommend 3 or more estimation methodologies --
“Triangulate” on your answer

- Want to use complementary ones
 - Do not select the same methods with the same biases.



Selecting methods

- Methods that would be considered complementary would be:
 - Top-down with bottom up
 - Expert opinion with Analogy
 - Expert opinion with Proxy Points
 - Expert opinions by experts with different project experiences and responsibilities
 - Proxy points and Analogy
- Methods that would be considered to have the same biases would be:
 - Algorithmic Models based on the same underlying algorithms (such as all of the COCOMO derivatives)
 - Experts who have had the similar responsibilities on similar projects^[1]

^[1] Jorgensen, Magne, Practical Guidelines for Better Software Estimation, 2005



Combining Estimates

□ Arithmetic Combinations

- Mean with SD
- Weighted Mean
 - Normal Distribution
 - Expected Validity



My results from student projects

- Estimations were combined – either thru averaging or other combination methods such as weighted averages
 - Outliers were ignored or lightly weighted
 - Using the combination techniques, their results converged

- NB. Even though they were estimating the same project, they typically defined the project differently, and so the groups never converged to 1 number



Use and Accuracy of Estimations

- The need for accuracy of an estimate is related to its purpose at that point in time.
- Feasibility stage:
 - need only be accurate enough to support a decision to prepare a more detailed definition of the project.
- Requirements specification
 - Critical project decision point
 - You are potentially committing significant resources based on the business case, which is based upon the cost estimate.
 - The required accuracy depends on the business model. (external vs. internal project...what are sensitivities)
- Detailed Design Complete:
 - hopefully most of the unknowns have become known, and a reasonably accurate estimate and plan can be created.



What do people actually do?

- Expert Estimation is most frequently used



One more issue: Target vs. Estimation



Vs.



This is a big question

Are you being told or asked?

Even if you are being told, then all of the same tools and techniques apply



When you are given an effort-budget

□ You then need to

- Size the job *as it is currently defined*
- Understand quantitatively the difference between the budget and what you estimate
- Work to bring the budget and your estimate into alignment



Aligning the budget and the estimate

□ Reducing your estimate by

○ Decreasing the cost drivers and risk factors...for example, by

- Simplifying design, reducing complexity, reusing components, omitting optional functionality etc. etc. etc.

○ Increasing productivity by

- Using more productive language, using higher skilled people, better technology, etc

□ Attempt to increase budget by quantitatively demonstrating the estimate



□ My Three Final Recommendations



The Golden Rule of Estimation

**Require all estimates to be
justified**

Gut feeling is not an adequate
justification



Do Not Use Tools or Methods Blindly

Try estimating previous projects to
validate/tune methods



Train your estimators

Knowing how to do something doesn't mean you know
how long it will take

Need to train estimators – accuracy correlated with
training and ability to see results (and responsibility),
rather than development experience





Questions?



Thank You!

