Fingers In The Air

A Gentle Introduction To Software Estimation

Giovanni Asproni
gasproni@asprotunity.com

Summary

- Some definitions
- What to estimate
- Estimation uncertainty
- Some estimation techniques
- Concluding remarks
- Questions

“Prediction is very difficult, especially about the future”

Niels Bohr

Question 1

Do you give estimates for your tasks?

Question 2

Do you negotiate your estimates?

Some Definitions: Estimate

An estimate is “an **approximate** calculation or judgement of the value, number, quantity, or extent of something”

The New Oxford Dictionary of English

Estimate: Examples

- Implementing the search functionality will require between two and four days
- The development costs will be between forty and sixty million pounds

Some Definitions: Target

A target is a statement of
a desirable business objective

Target: Examples

- Release version 1.0 by Christmas
- The system must support at least 400 concurrent users
- The cost must not exceed three million pounds

Some Definitions: Commitment

A commitment is a promise to deliver specified functionality at a certain level of quality by a certain date

Commitment: Examples

- The search functionality will be available in the next release
- The response time will be improved by 30% by the end of the next iteration

Estimates, Targets, and Commitments

- Estimates, targets, and commitments are **independent** from each other
- But you better base your targets and commitments on sound estimates

Purpose Of Estimation (1/3)

“The primary purpose of software estimation is **not to predict a project’s outcome**; it is to **determine** whether a project’s **targets** are **realistic enough** to allow the project to be controlled to meet them”

Steve McConnell, Software Estimation

Purpose Of Estimation (2/3)

- Is to make proper project management and planning possible
  - Allowing the project stakeholders to make commitments based on realistic targets

Purpose Of Estimation (3/3)

Estimates are **not** negotiable

What To Estimate?

- Any factor important for the project success
  - Size
  - Cost
  - Effort
  - Time
  - Risk
  - Etc.

Accuracy And Precision

- Often used (wrongly) as synonyms
- A measurement can be
  - Precise without being accurate
  - Accurate without being precise

Accuracy

- Accuracy refers to how close to the real value a number is
- Example
  - 2.7 is an accurate representation of e to two significant digits (e is the mathematical constant equal to 2.718281...)

Precision

- Precision refers to how exact a number is
- Example
  - 2.8183 is a precise representation for e
  - It is less accurate than 2.7

Accuracy Or Precision?

- Estimates should be **accurate** not **precise**
  - Accurate: task x will take between 2 and 4 days
  - Precise: task x will take 3.04 days

Accuracy And Precision

- Always match the precision of your estimate to its accuracy
  - Use appropriate units of measure
  - E.g., do not use hours to estimate years worth of work

Estimating Software Is Hard (1/2)

- Differences in individual productivity
- Creative processes are difficult to plan
- People can only think so fast
- Software is intangible
  - Difficult to measure

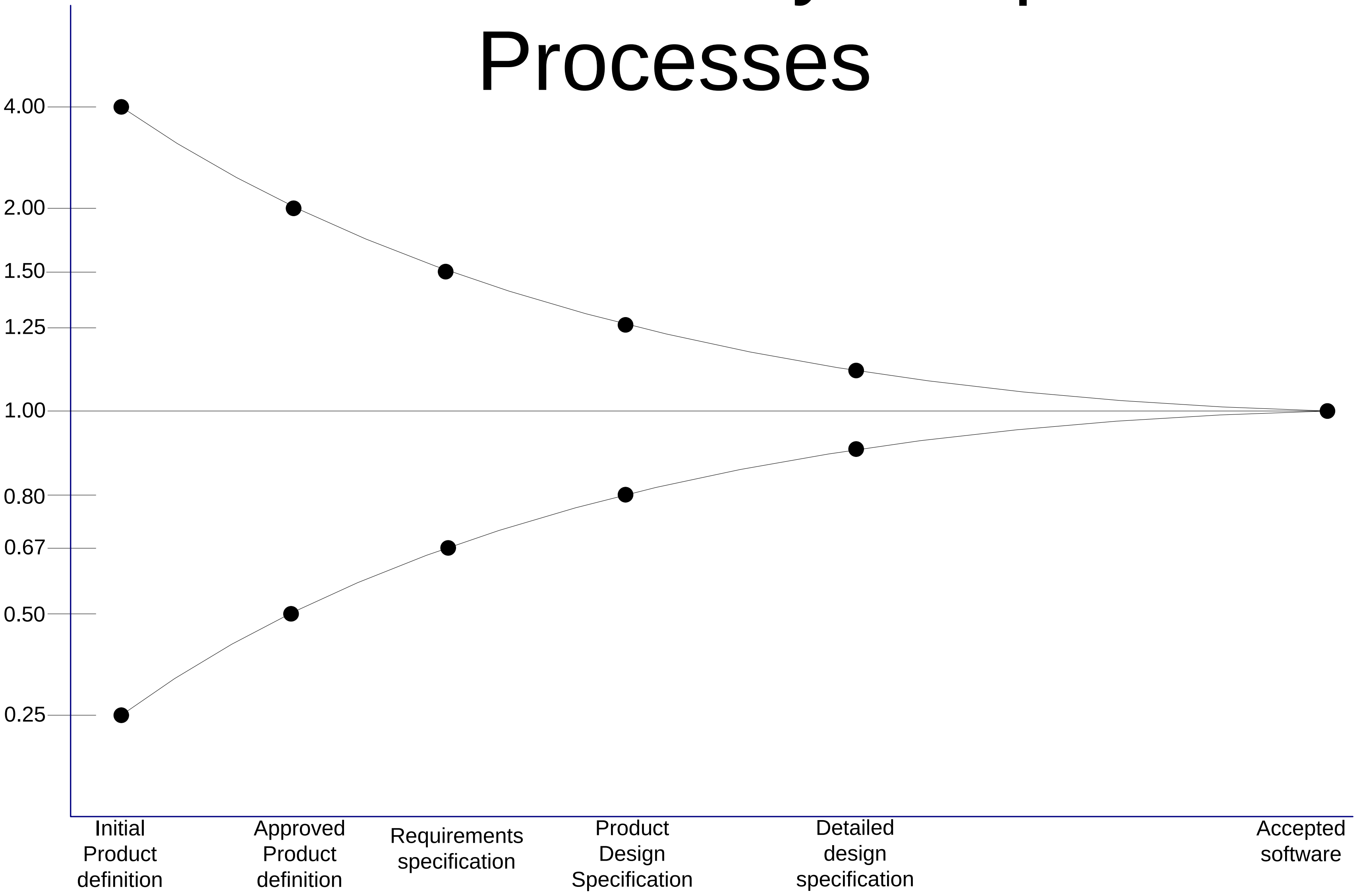
Estimating Software Is Hard (2/2)

- Estimation errors happens because of
  - Omissions
  - Uncertainty
  - Change

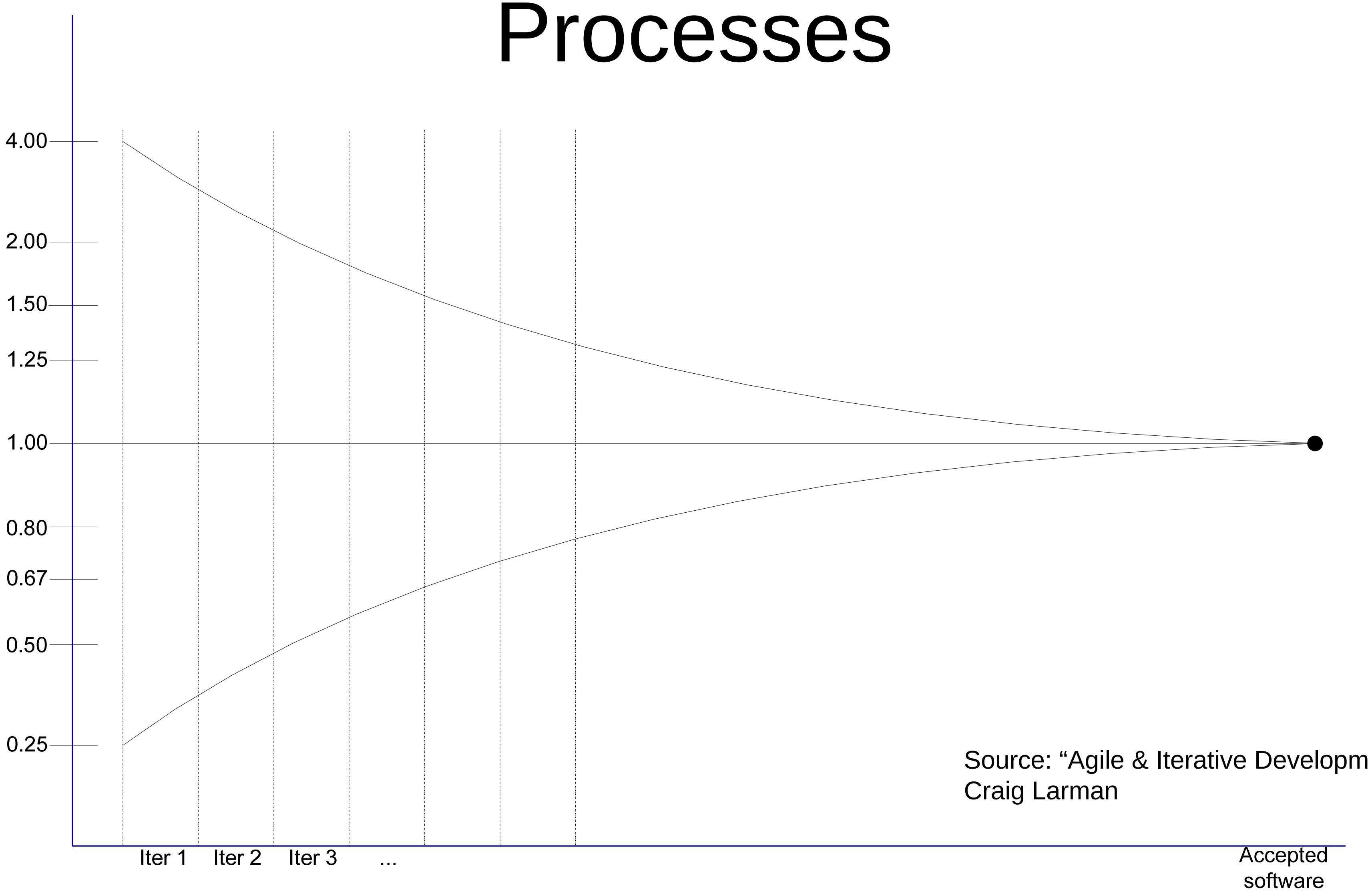
Rule Of Estimation Number 1
(By Paul Coombs)

“Your estimate will be wrong”

Cone Of Uncertainty: Sequential Processes



Cone Of Uncertainty: Iterative Processes

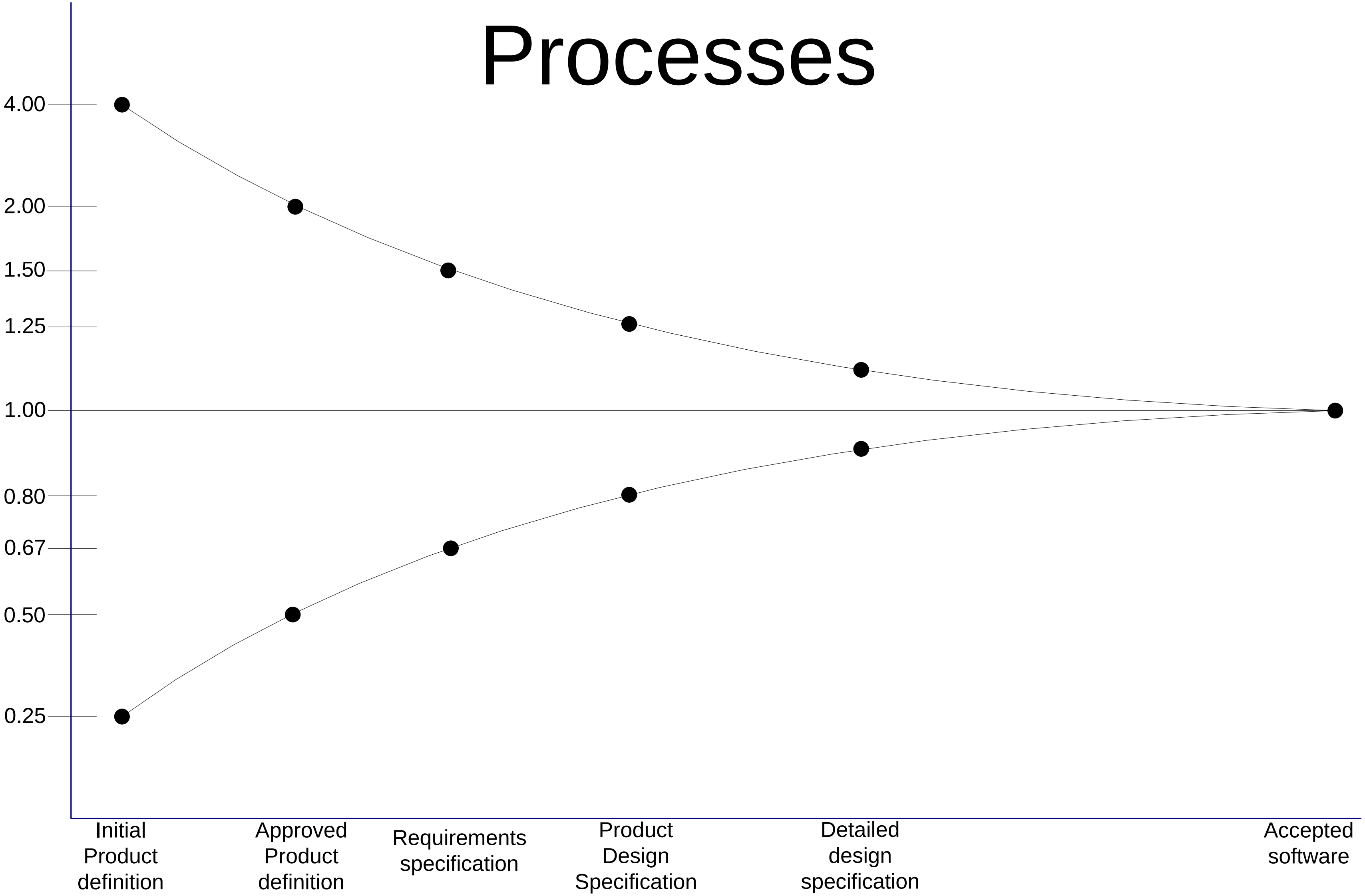


Source: “Agile & Iterative Development”
Craig Larman

Cone Of Uncertainty: A Few Considerations

- It represents **best case accuracy**
  - You cannot beat the cone
- The cone doesn’t narrow itself
  - The project needs to be managed properly

Cone Of Uncertainty: Sequential Processes



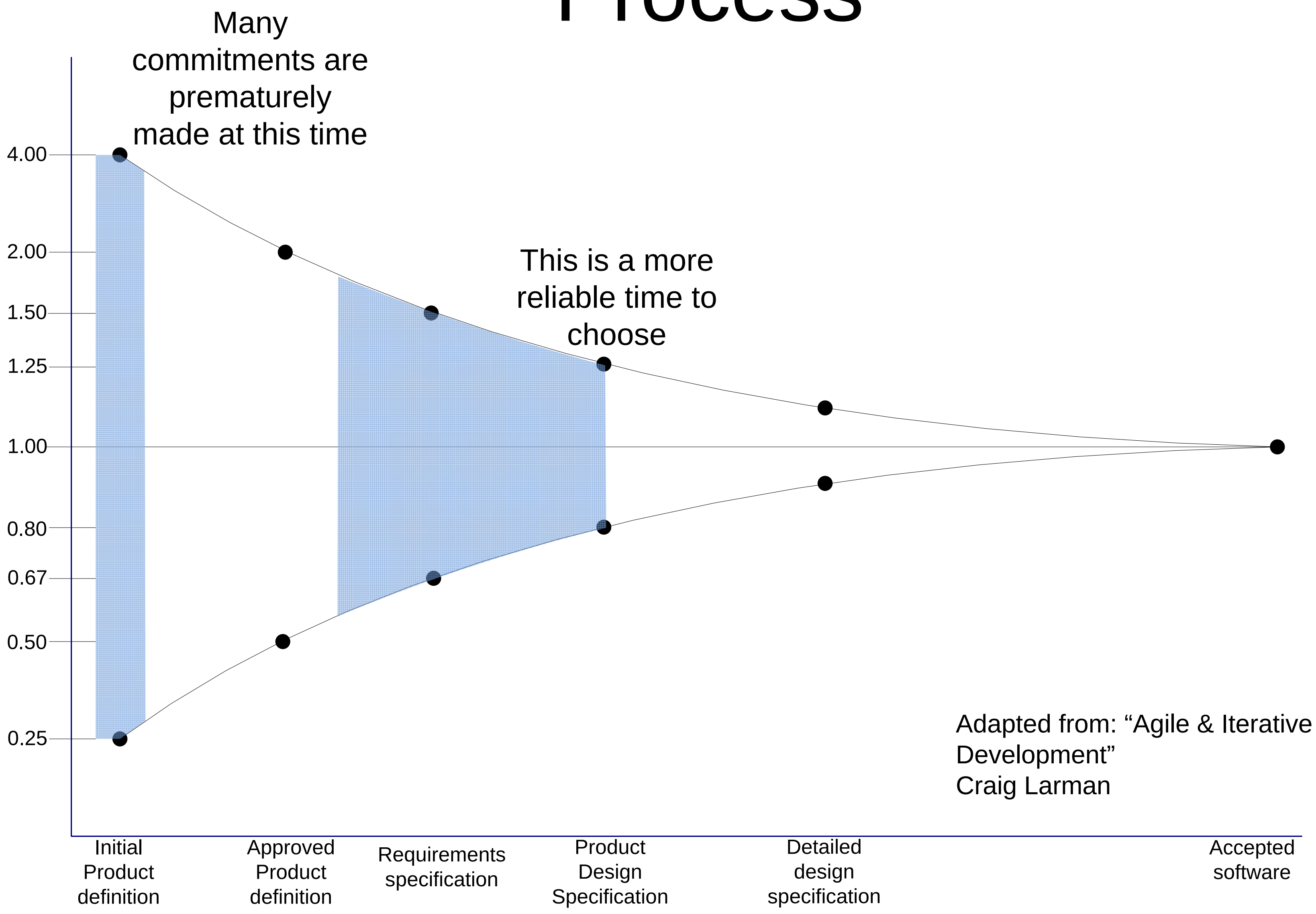
Warning!

- If your project has one or more of
  - Poor requirements definition
  - Lack of user involvement
  - Poor design
  - Poor coding practices
  - Bad planning
  - Etc.
- Then estimates cannot save you!

Estimation Is An On-Going Activity

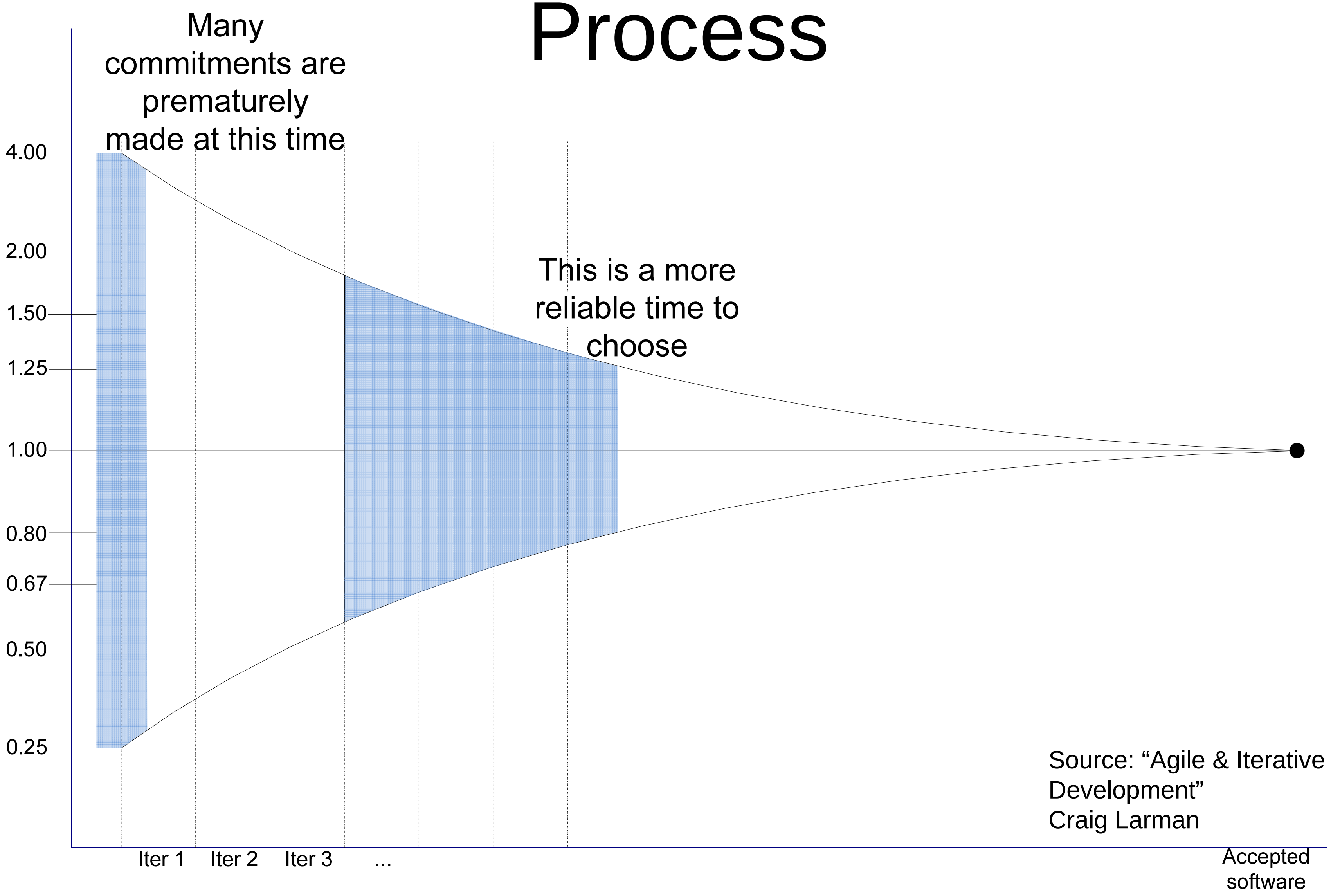
- Estimates should change whenever new knowledge is available
  - This implies that planning should also be an on-going activity

When To Commit?: Sequential Process



Adapted from: “Agile & Iterative Development”
Craig Larman

When To Commit?: Iterative Process



Source: “Agile & Iterative Development”
Craig Larman

Overestimation And Underestimation

- Accurate estimates are rare
- Is it better to overestimate or underestimate?

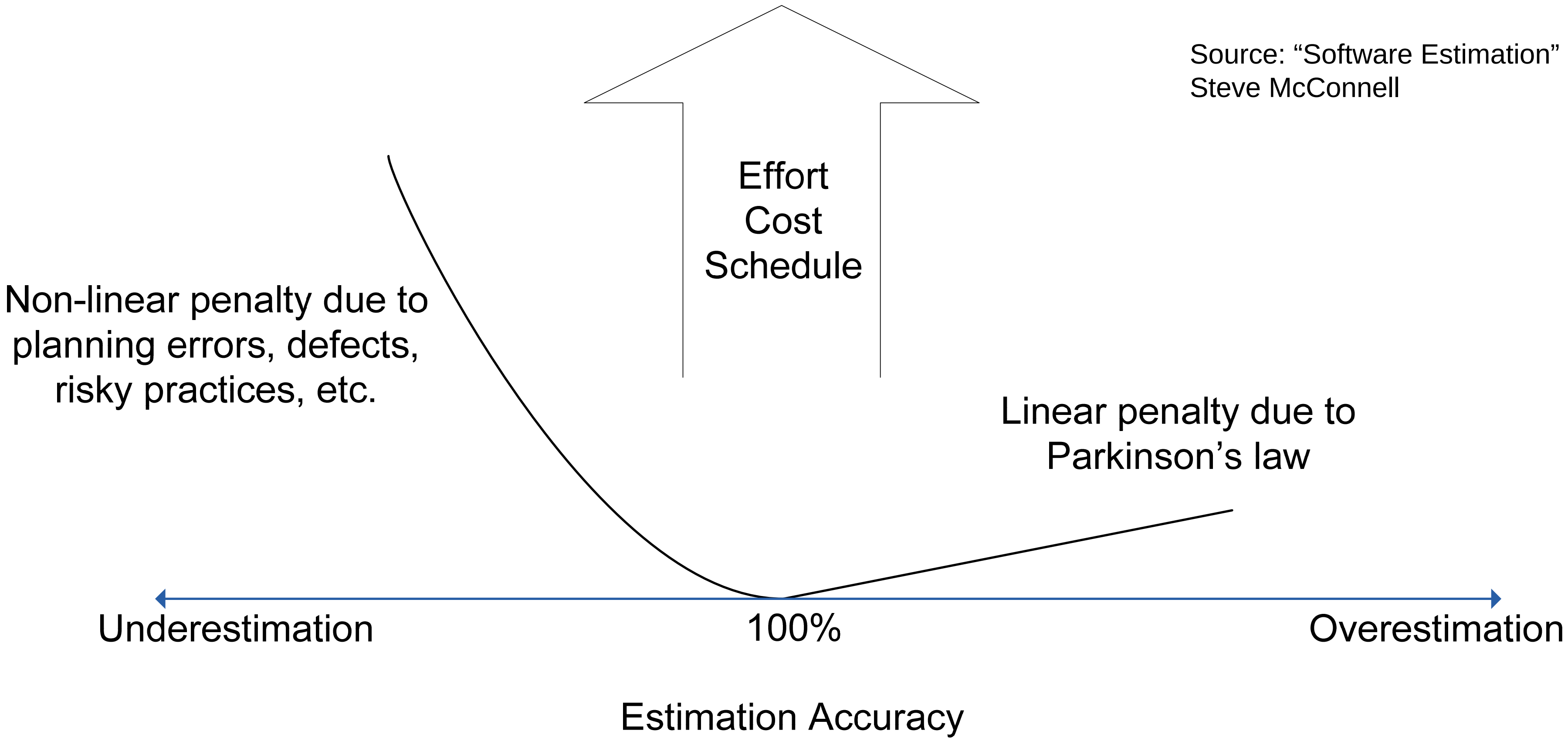
Problems Due To Underestimation

- Reduced effectiveness of planning
- Reduced chance of on-time delivery
  - Developers tend to be 20%-30% too optimistic
- Increases the chance of cutting corners
- Destructive project dynamics
  - Overtime
  - Angry customers
  - Poor quality

Problems Due To Overestimation

- Parkinson’s law
  - Work will expand to fill available time
- Student’s syndrome
  - If developers are given too much time, they’ll procrastinate until late in the project, and probably they won’t finish on time

Underestimation And Overestimation Effects


Effort
Cost
Schedule
Non-linear penalty due to planning errors, defects, risky practices, etc.
Linear penalty due to Parkinson's law
Underestimation
100%
Overestimation
Estimation Accuracy

Source: “Software Estimation”
Steve McConnell

Estimation Techniques Fundamentals (1/5)

- All based on historical data
  - Industry
  - Company
  - Current project

Estimation Techniques Fundamentals (2/5)

- Uncertainty is managed using probabilities
  - An Estimate has always a probability attached
  - Single point estimates always have a probability less than 100%
  - The best you can do is give a best, a worst, and a likely outcome

Estimation Techniques Fundamentals (3/5)

- Measurement is important
  - It’s difficult to estimate something you can’t measure

Estimation Techniques Fundamentals (4/5)

- Impossible to estimate...
  - The system must be scalable to support future business growth
- ...Much better
  - The system must be able to support 400 concurrent users with a maximum response time of 1 second

Estimation Techniques Fundamentals (5/5)

- Let the people who do the work create the estimates
  - Developers tend to be accurate...
  - ...when they remember to include everything!

Often Forgotten Things

- Include all the tasks (build environment, machine setup, supporting the build, etc.)
- Keep into account time spent on side tasks (e-mail, meetings, phone calls, etc.)
- Plan for absence (holidays, illness, etc.)
- Make the underlying assumptions explicit

Some Estimation Drivers

- Product size
- Product complexity
- Human factors
- Previous experience
- Available tools (programming languages, etc.)

Some Estimation Techniques

- Count, Compute, Judge
- Mathematical models
- Calibration and historical data
- Analogy
- Proxy
- Expert judgement

Count, Compute, Judge

- Count if possible
- Compute if you can’t
- Use judgment as a last resort

Mathematical Models (1/2)

- Based on formulas obtained by studying the data available from past projects
- Pros
  - – Software tools available
- Cons
  - – Too many available “knobs”
  - – Accuracy is not better than other techniques

Mathematical Models (2/2)

Example: COCOMO II

$$PM_{NS} = A \times Size^{E} \times \prod_{i=1}^{n} EM_{i}$$

$$E = B + 0.01 \times \sum_{j=1}^{5} SF_{j}$$

$$TDEV_{NS} = C \times (PM_{NS})^{F}$$

$$F = D + 0.2 \times (E - B)$$

$$A = 2.94$$

$$B = 0.91$$

$$C = 3.67$$

$$D = 0.28$$

Calibration And Historical Data (1/2)

- Estimates based on available data on past performance
- Pros
  - Avoids subjectivity and wishful thinking
  - High accuracy
  - Data can be used to calibrate software estimation tools
- Cons
  - Need the data to start with

Calibration And Historical Data (2/2)

- Data to collect
  - Size
  - Effort
  - Time
  - Defects

Estimation By Analogy

- Comparison to similar past projects
  – Similar functionality
  – Similar number of classes
  – Etc.
- Pros
  – Simple to implement
- Cons
  – Subjectivity
  – Less accurate than other techniques

Estimation By Proxy

- Find a proxy correlated to what you are interested on, but is easier to estimate
  - Story points
- Pros
  - High accuracy possible
- Cons
  - High discipline and experience required

Decomposition And Recomposition

- Decompose large pieces into smaller chunks and combine the individual estimates into an aggregate
- Pros
  - High accuracy possible
  - Simple to understand and apply
- Cons
  - Be careful: the aggregation of best/worst cases is not a simple sum

Expert Judgement (1/2)

- Individual judgement
- Group techniques
  - Average of estimates
  - Delphi
  - Planning poker

Expert Judgment (2/2)

- By far the most used technique
- Pros
  - – High accuracy possible
- Cons
  - – External pressure to cut estimates
  - – Needs experts

Putting Everything Together (1/4)

- The different techniques can (and should) be used at the same time

Putting Everything Together (2/4)

- If you have historical data, start with it
  - Assume that productivity will be the same
  - Avoid wishful thinking (we’ll do it better this time syndrome)
- Use the other techniques to double check the estimates
- Use estimation tools if available

Putting Everything Together (3/4)

- If you do not have historical data
  - Look for industry data
  - Use expert judgment, estimation by proxy, and estimation by analogy

Putting Everything Together (4/4)

- In any case
  - Keep track of your original estimates and the actual results
  - Collect the data and use for the rest of the project (and future ones as well)

Conclusion (1/3)

- When creating estimates
  - Make sure you distinguish them from targets and commitments
  - Let people who do the work create them
  - Make the underlying assumptions explicit
  - Be careful with under-estimation
  - Remember that estimates are **not** negotiable!

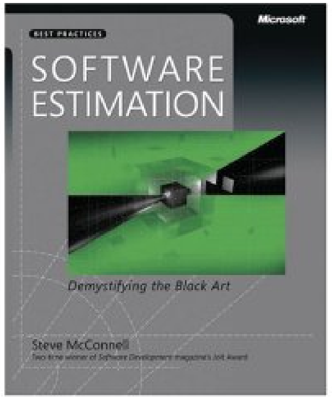
Conclusion (2/3)

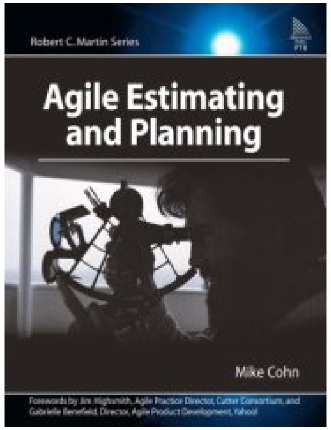
- When estimating remember to
  - Include all the tasks (build environment, machine setup, supporting the build, etc.)
  - Plan for absence (holidays, illness, etc.)
  - Keep into account time spent on side tasks (e-mail, meetings, phone calls, etc.)

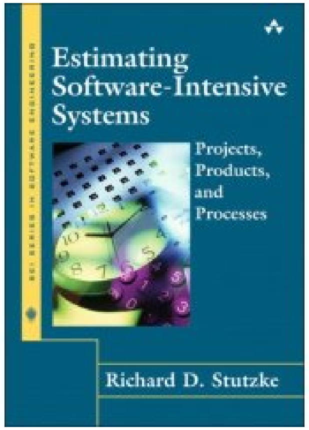
Conclusion (3/3)

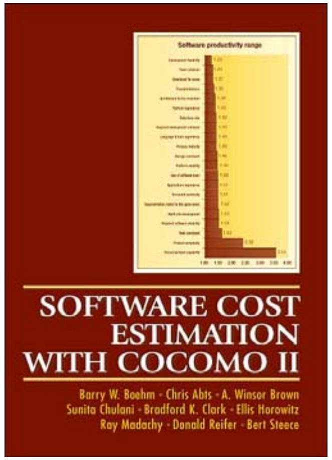
- Record project data
- Use more than one estimation technique
- Estimation is an ongoing activity
  - Refine the estimates as you go
- Your estimates will be wrong!

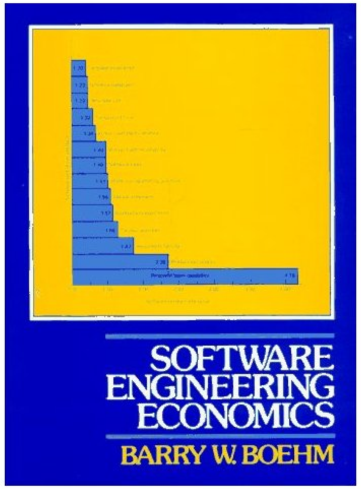
References

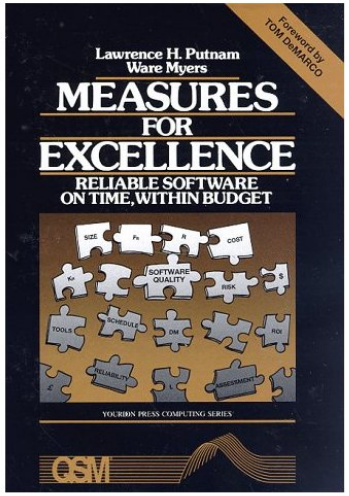

BEST PRACTICES
Microsoft
SOFTWARE ESTIMATION
Demystifying the Black Art
Steve McConnell

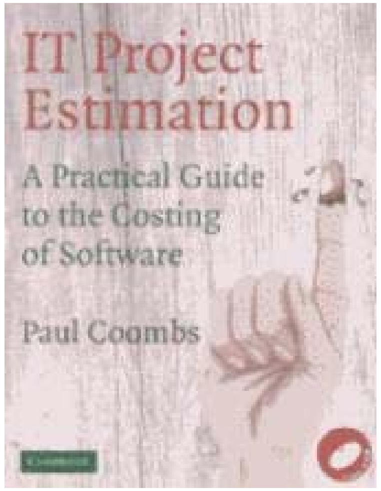

Robert C. Martin Series
Agile Estimating and Planning
Mike Cohn

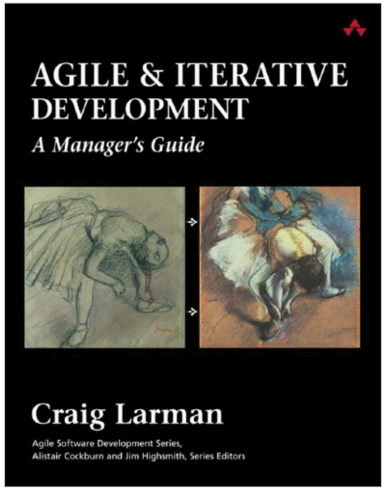

SEI SERIES IN SOFTWARE ENGINEERING
Estimating Software-Intensive Systems
Projects, Products, and Processes
Richard D. Stutzke


Software productivity range
SOFTWARE COST ESTIMATION WITH COCOMO II
Barry W. Boehm · Chris Abts · A. Winsor Brown
Sunita Chulani · Bradford K. Clark · Ellis Horowitz
Ray Madachy · Donald Reifer · Bert Steece


SOFTWARE ENGINEERING ECONOMICS
BARRY W. BOEHM


Foreword by TOM DeMARCO
Lawrence H. Putnam
Ware Myers
MEASURES FOR EXCELLENCE
RELIABLE SOFTWARE ON TIME, WITHIN BUDGET
SIZE
COST
SOFTWARE QUALITY
RISK
TOOLS
SCHEDULE
RELIABILITY
ASSESSMENT
ROI
YOURDON PRESS COMPUTING SERIES
QSM


IT Project Estimation
A Practical Guide to the Costing of Software
Paul Coombs
CAMBRIDGE


AGILE & ITERATIVE DEVELOPMENT
A Manager's Guide
Craig Larman
Agile Software Development Series,
Alistair Cockburn and Jim Highsmith, Series Editors

Questions?