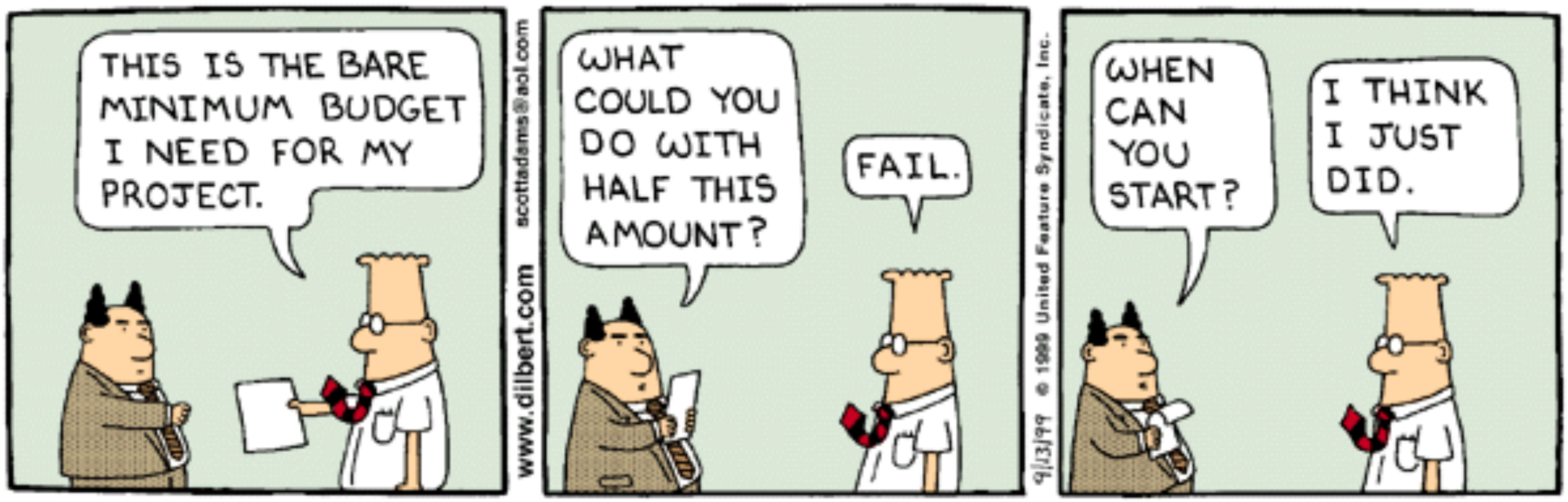
Dilbert on Cost Estimation

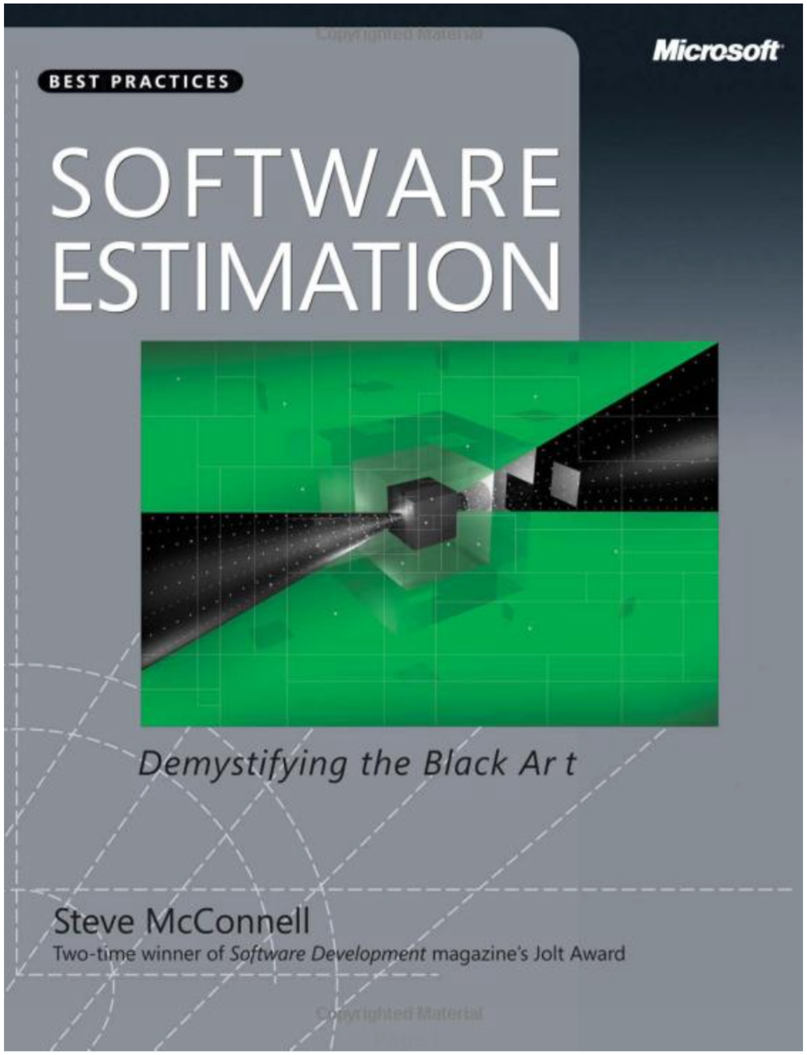

THIS IS THE BARE MINIMUM BUDGET I NEED FOR MY PROJECT.
WHAT COULD YOU DO WITH HALF THIS AMOUNT?
FAIL.
WHEN CAN YOU START?
I THINK I JUST DID.
www.dilbert.com
scottadams@aol.com
9/13/99 © 1999 United Feature Syndicate, Inc.

SE463

Software Requirements Specification & Analysis

Software Estimation

Sources


Microsoft
BEST PRACTICES
SOFTWARE ESTIMATION
Demystifying the Black Ar t
Steve McConnell
Two-time winner of Software Development magazine's Jolt Award
Copyrighted Material

Lecture content comes from

Steve McConnell, *Software Estimation: Demystifying the Black Art*, Microsoft Press, 2006.

Why Estimate Software Cost and Effort?

- To assess economic feasibility
- To provide a basis for agreeing to a job
- To make commitments that we can meet
- To understand resource needs

Terminology

An estimate is a prediction of how long a project will take or how much it will cost.

A target is a statement of a desirable business objectives.

A commitment is a promise to deliver.

Steve McConnell, *Software Estimation: Demystifying the Black Art*, Microsoft Press, 2006.

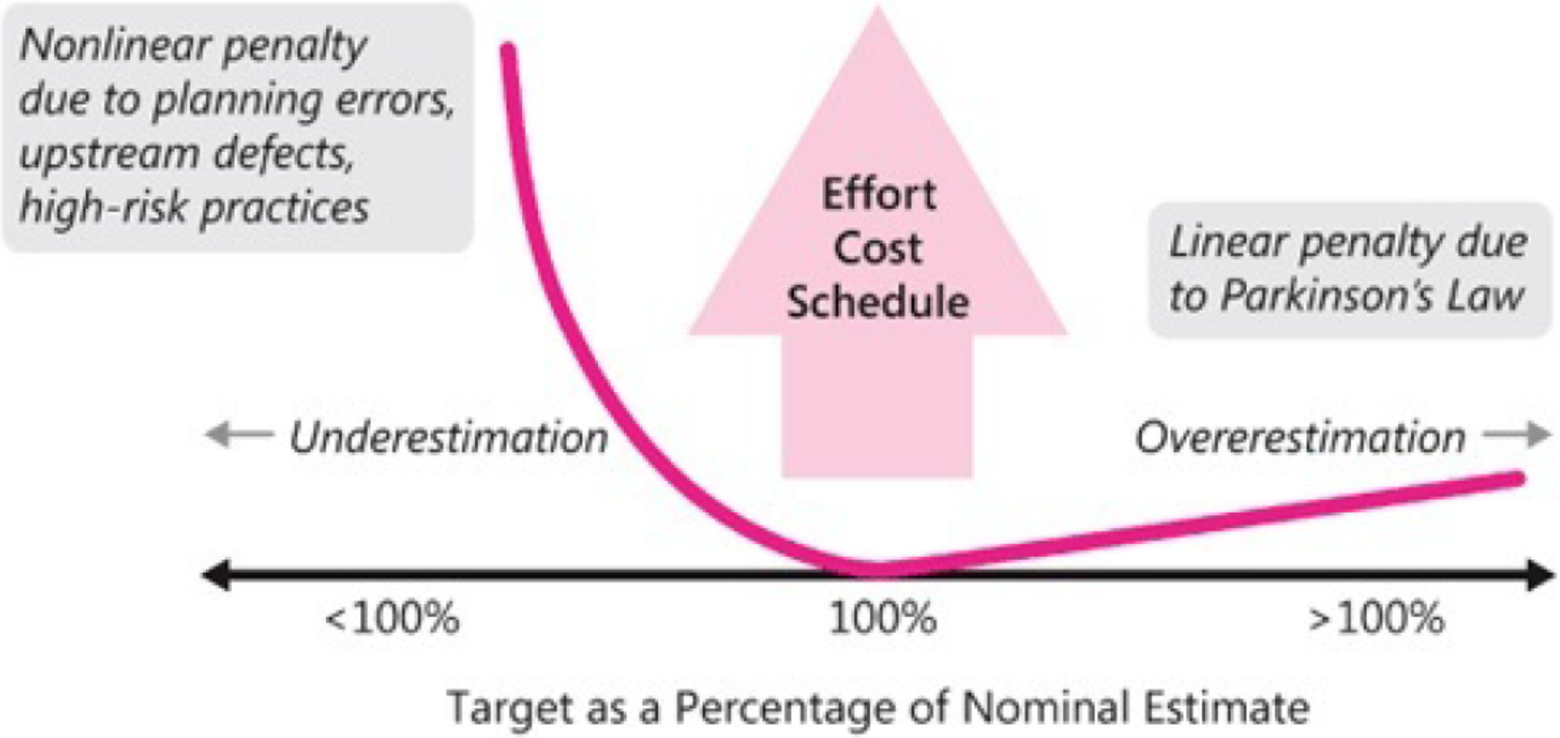
Inaccurate Estimates

"An estimate is the most optimistic prediction that has a non-zero probability of coming true.

Accepting this definition leads irrevocably toward a method called what's-the-earliest-date-by-which-you-can't-prove-you-won't-be-finished estimating."

Tom DeMarco

Effects of Inaccurate Estimates



Steve McConnell, *Software Estimation: Demystifying the Black Art*, Microsoft Press, 2006.

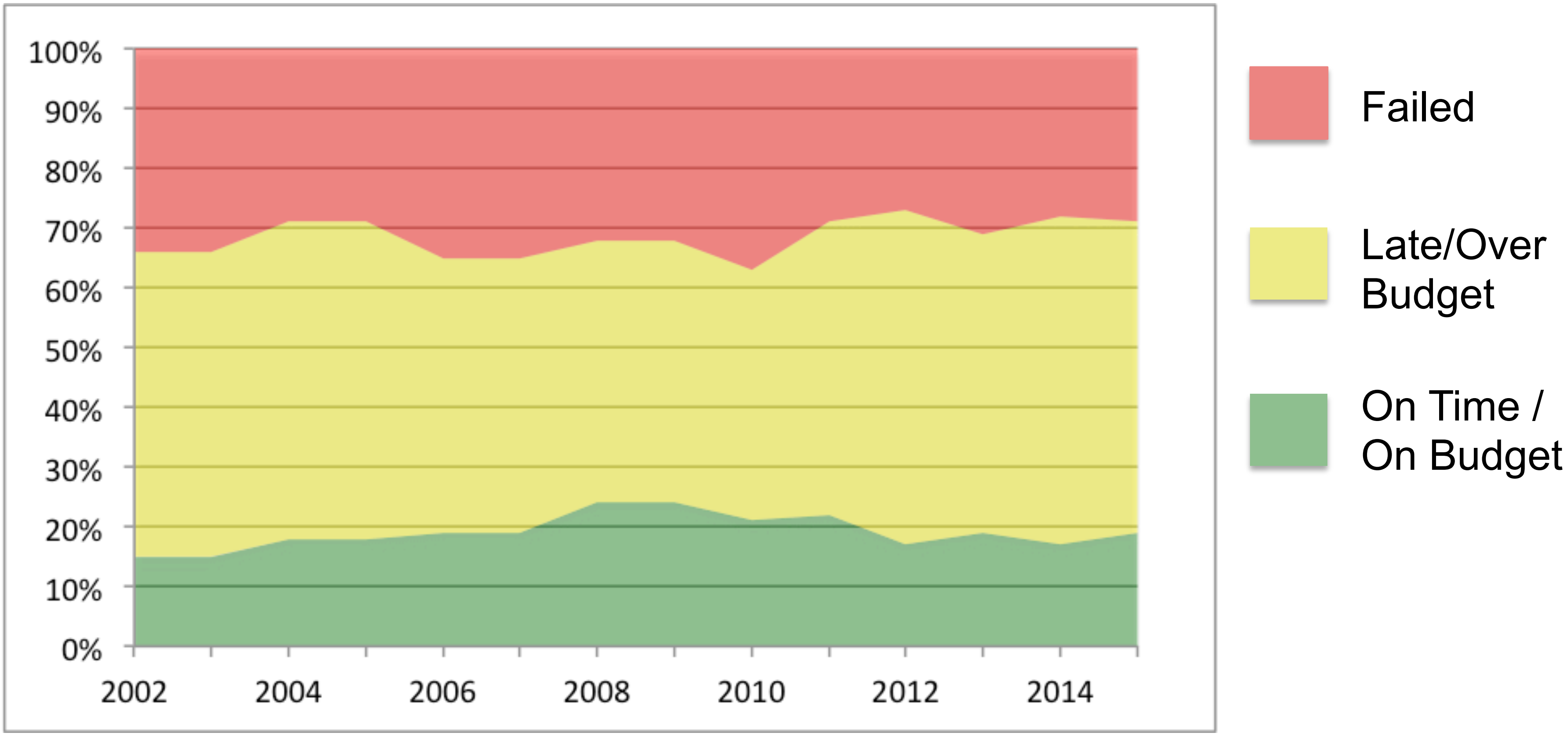
Software Estimation

How long (hours) would it take you to implement a Stack ADT (push, pop, top)?

How long would it take you to re-implement your OS project?

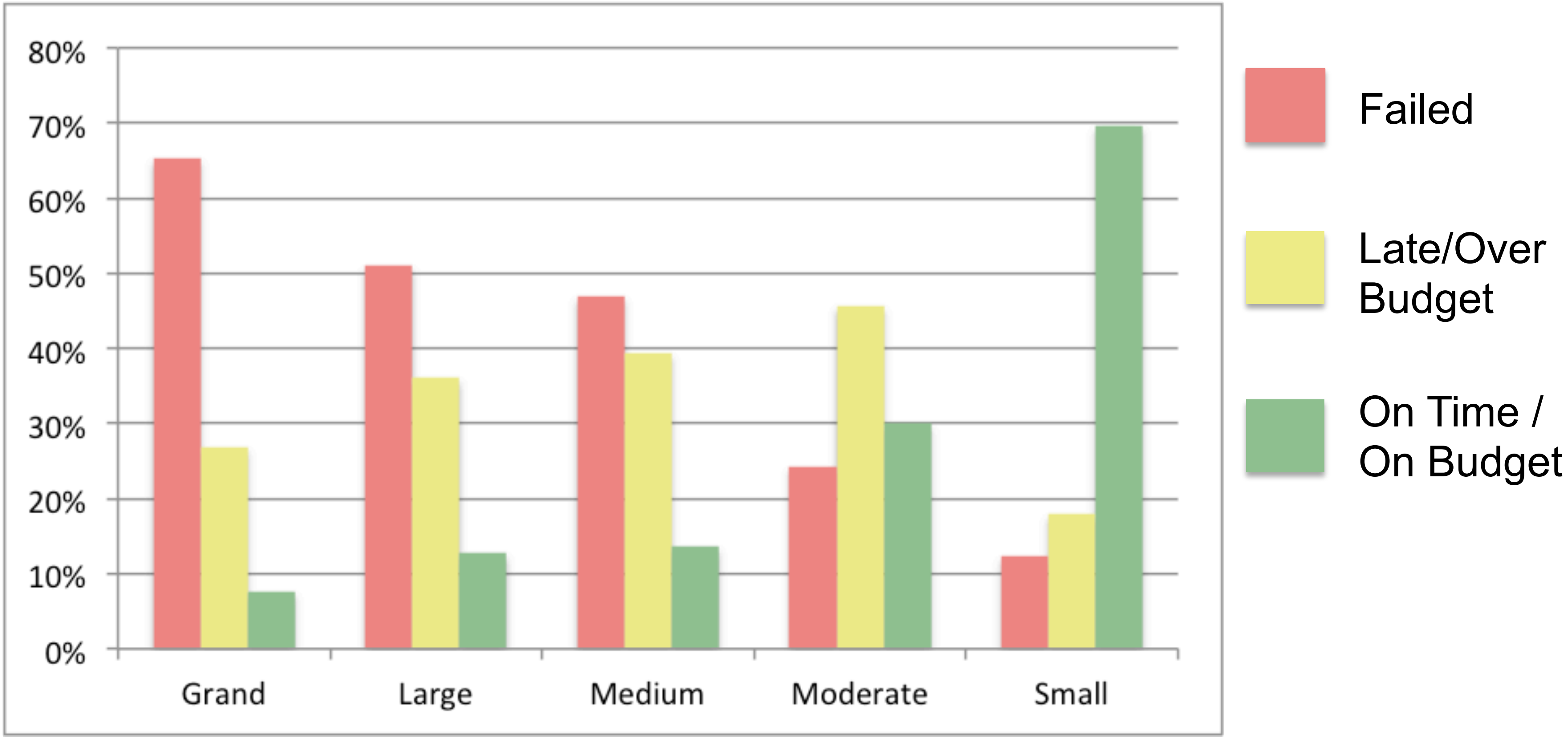
How long would it take you to implement your SE463 project?

Industry's Track Record



The Standish Group, The CHAOS Report, 2002-2015

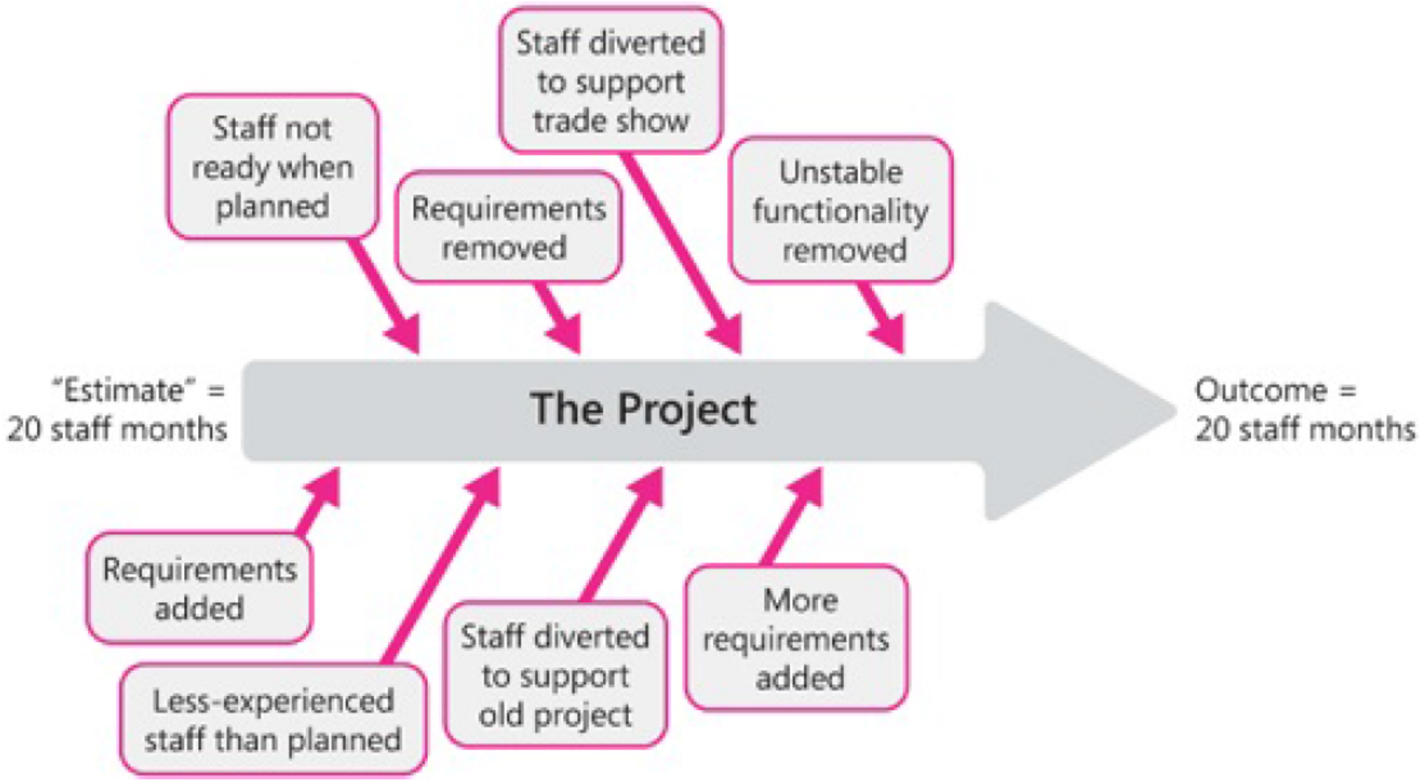
Industry’s Track Record



The Standish Group, The CHAOS Report, 2002-2015

What is a Good Estimate?

A project is successful if it is delivered on time, within budget, with acceptable functionality.


Staff not ready when planned
Requirements removed
Staff diverted to support trade show
Unstable functionality removed
"Estimate" = 20 staff months
The Project
Outcome = 20 staff months
Requirements added
Less-experienced staff than planned
Staff diverted to support old project
More requirements added

Steve McConnell, *Software Estimation: Demystifying the Black Art*, Microsoft Press, 2006.

What is a Good Estimate?

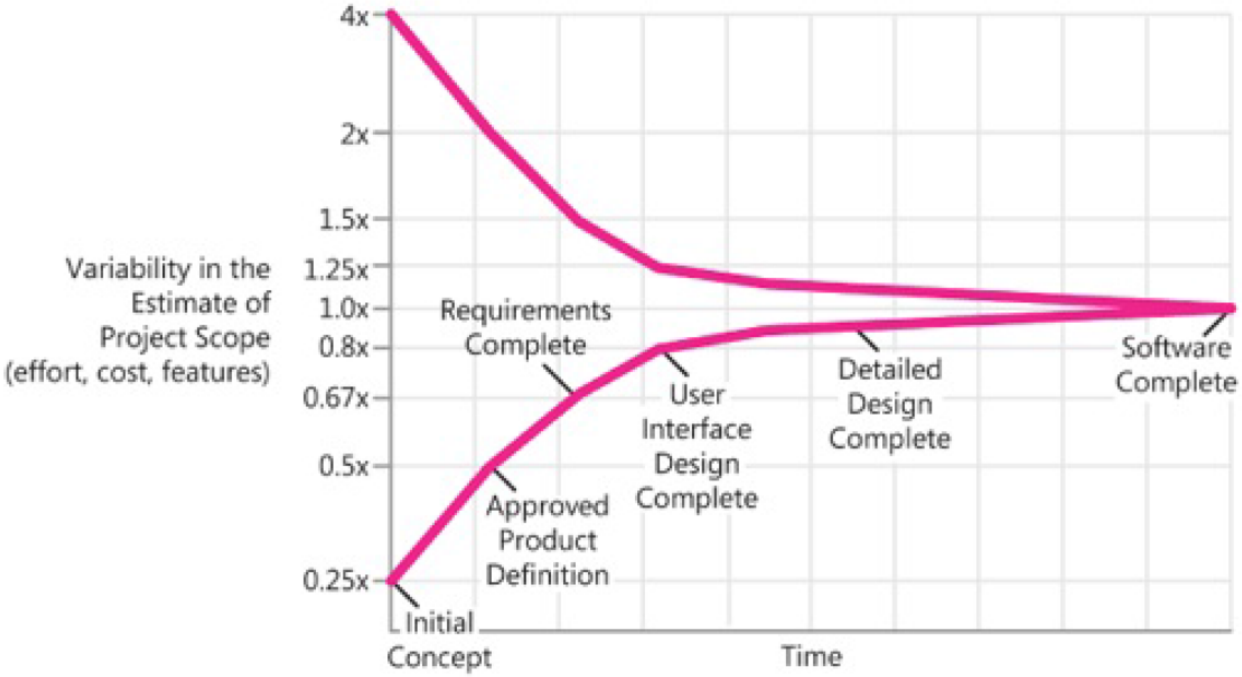
Once we make an estimate, and make a commitment to deliver functionality and quality by a particular date, then we control the project to meet the target.

- remove noncritical requirements
- redefine requirements
- replacing less-experienced staff with more-experienced staff

If the initial target and initial estimate are within about 20% of each other, the project manager should have enough maneuvering room to meet the project's business goals.

Steve McConnell, *Software Estimation: Demystifying the Black Art*, Microsoft Press, 2006.

Sources of Estimation Error



Uncertainty about how numerous requirements details and design decisions will be made.

Estimation accuracy improves rapidly for the first 30% of the project.

Steve McConnell, *Software Estimation: Demystifying the Black Art*, Microsoft Press, 2006.

Sources of Estimation Error

Omitted Activities

- Omitted Functional / Quality Requirements
  - Initialization, data conversion, glue code
- Software Development Activities
  - Integration, test data, performance tuning, technical reviews
- Non-Software Development Activities
  - Vacations, sick days, training, company meetings

Steve McConnell, *Software Estimation: Demystifying the Black Art*, Microsoft Press, 2006.

Sources of Estimation Error

Optimism

- Developer estimates tended to contain an optimism factor of 20% to 30%
- Managers believe that projects can be completed 30% faster than previous projects
    - Team more productive, less will go wrong, no learning curve

Bias

- Managers who want estimates to align with targets will put pressure on estimates, schedules, project teams.

Subjectivity

- Some estimation techniques include multiple adjustment knobs that allow subjectivity to creep in.

Steve McConnell, *Software Estimation: Demystifying the Black Art*, Microsoft Press, 2006.

Biggest Influences on Estimates

Project Size

- The largest driver in a software estimate is the size of the software being built
- Software projects have diseconomies of scale: larger projects require more coordination and communication

Kind of Software

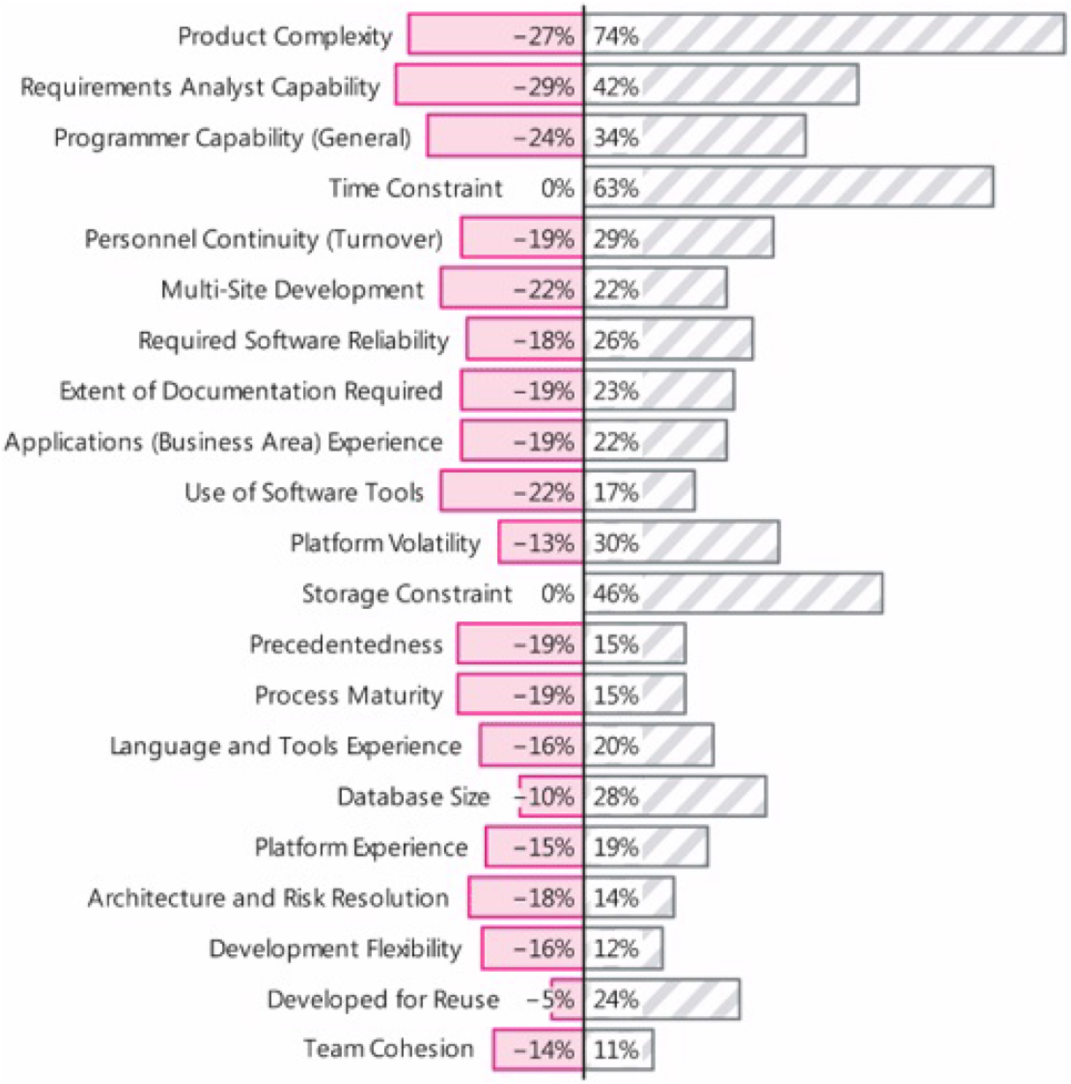
- Business applications, embedded systems, real-time

Personnel Factors

- Productivity of individuals can vary by a factor of 10 or more
- Mostly accounted for if estimates are based on team’s historical performance

Steve McConnell, *Software Estimation: Demystifying the Black Art*, Microsoft Press, 2006.

Other Influences on Estimates



Steve McConnell, *Software Estimation: Demystifying the Black Art*, Microsoft Press, 2006.

Data to Collect

Whenever possible count or compute your estimate rather than using expert judgment.

In order to compute software estimates based on historical data, we need to be collecting data about our projects.

- Project size
  - lines of code (LOC)
  - requirements
    - e.g., function points, scenarios, stories, UI screens
- Effort (staff months)
- Time (calendar months)

Collecting data on size and effort every 1 to 2 weeks can provide valuable insight into your project's dynamics.

Steve McConnell, *Software Estimation: Demystifying the Black Art*, Microsoft Press, 2006.

Calibration

Compute estimates of new development tasks based on the actual work to complete past tasks.

- Project data - data generated earlier in the same project that’s being estimated
- Historical data - data from the organization that will conduct the project being estimated
- Industry data - data from other organizations that develop the same basic kind of software as the software that’s being estimated

Steve McConnell, *Software Estimation: Demystifying the Black Art*, Microsoft Press, 2006.

Estimation by Analogy

Compute a size estimate based on a piece-by-piece count of analogous elements, and the past sizes of those elements.

Subsystem	Old Project (Size)	New Project (Est. Size)	Multiplier	Old Project (LOC)	New Project (LOC)
Use Cases	10 use cases	14 use cases	1.4	5,000	7000
User Interface	14 screens	16 screens	1.1	14,000	15,400
Graphs	10 graphs	16 graphs	1.6	9,000	25,600
Reports	3 reports	5 reports	1.7	3,000	5,100
Business Rules	???	???	1.5 (?)	11,000	16,500
TOTAL				**42,000**	**69,600**

Historical data
Estimates
Computed values

Steve McConnell, *Software Estimation: Demystifying the Black Art*, Microsoft Press, 2006.

Commercial Estimation Tools

Commercial tools embed computationally intensive estimation methods that cannot be done easily by hand or with a calculator.

- Simulating project outcomes (with respect to probability distributions for variability in system size, productivity)
- Probability analysis of estimates
- Complex effort estimations that account for diseconomies of scale

Tools tend to have large databases of industry-average data, but they can be calibrated with your own historical data.

Construx Estimate: www.construx.com/Resources/Construx_Estimate

Steve McConnell, *Software Estimation: Demystifying the Black Art*, Microsoft Press, 2006.

Function Point Analysis

Goal: To estimate cost based on what we know at requirements time

We break the problem down into three parts:

1. Estimate the number function points from the requirements
2. Estimate code size from function points
3. Estimate resources required (time, personnel, money) from code size

Steve McConnell, *Software Estimation: Demystifying the Black Art*, Microsoft Press, 2006.

Function Points

A function point is a size metric of functionality. It relates to the requirements of the software under development.

The number of function points in a system is based on the number and complexity of the following

- **External Inputs** – screens, forms, dialog boxes, messages
- **External Outputs** – screens, reports, graphs, messages
- **External Queries** – responses to input requests that do not change system data
- **Internal Logical Files** – major internal data stores (end-user data, control information)
- **External Interfaces / APIs** – files or APIs controlled by adjacent systems

Steve McConnell, *Software Estimation: Demystifying the Black Art*, Microsoft Press, 2006.

Function Point Counting

A function point count is computed by adding up inputs and outputs, weighted by complexity multipliers.

Program Characteristic	Low Complexity	Medium Complexity	High Complexity
External Inputs	___ x 3	___ x 4	___ x 6
External Outputs	___ x 4	___ x 5	___ x 7
External Queries	___ x 3	___ x 4	___ x 6
Internal Files	___ x 4	___ x 10	___ x 15
External Interfaces / APIs	___ x 5	___ x 7	___ x 10

Steve McConnell, *Software Estimation: Demystifying the Black Art*, Microsoft Press, 2006.

Estimating Code Size from FPs

Language	Average	Median	Low	High
ASP	51	54	15	69
C	97	99	39	333
C++	50	53	25	80
C#	54	59	29	70
Excel	209	191	131	315
HTML	34	40	14	48
J2EE	46	49	15	67
Java	53	53	14	134
JavaScript	47	53	31	63
.NET	57	60	53	60
Oracle	37	40	17	60
SAS	38	37	22	55
SQL	21	21	13	37

http://www.qsm.com/resources/function-point-languages-table

Structured Expert Judgment

Use expert judgment as a last resort, or for a second or third estimation.

- People doing the work will create the most accurate estimates
- It's best to decompose estimation – and estimate tasks that require no more than 2 days of effort
- It's best to estimate a range (best case vs. worst case) of sizes
  - Single-point estimates tend to be much closer to best-case estimates
- Program Evaluation and Review Technique (PERT) Formula

ExpectedCase = [BestCase + (4*MostLikelyCase) + WorstCase] /6

Steve McConnell, *Software Estimation: Demystifying the Black Art*, Microsoft Press, 2006.

Summary of Size Estimation Tech.

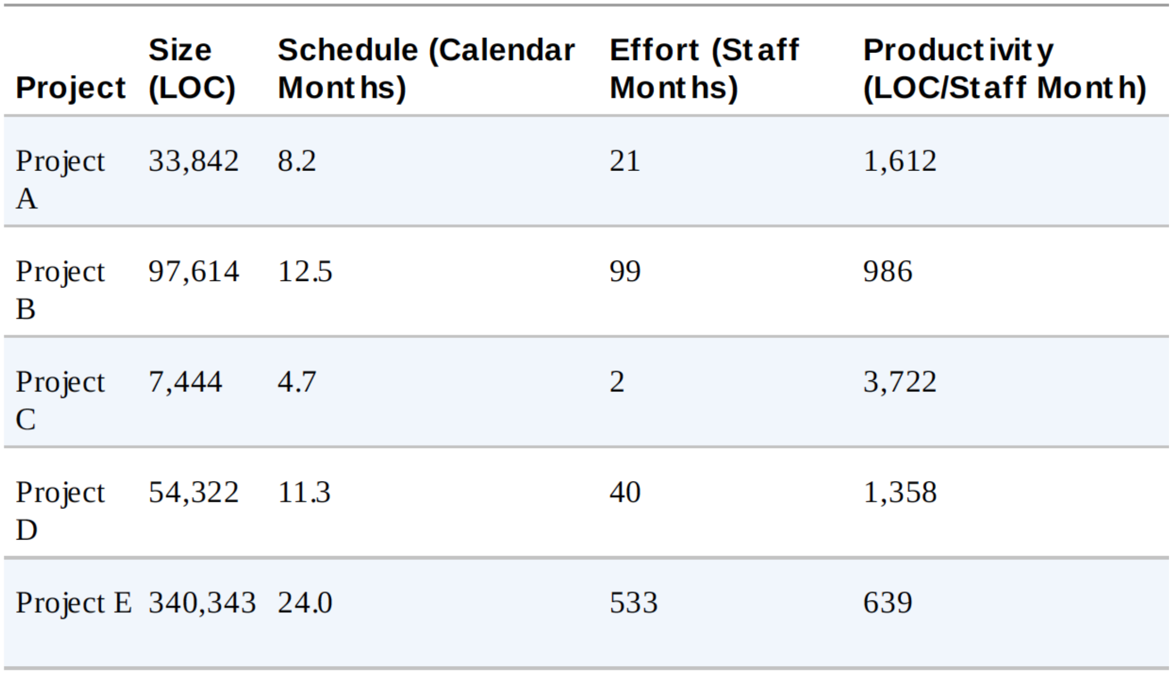
Technique	Kinds of Sizes that can be Estimated
Analogy	features, function points, screens, GUI components, LOC
Estimation Tools	function points, LOC
Function Point Analysis	function points, LOC
Structured Judgment	features, user stories, requirements, use cases, function points, Web pages, GUI components, database tables, interface definitions, classes, functions/subroutines, LOC
GUI Elements	function points, LOC
Standard Components	function points, LOC
Story Points	story points, LOC
Wideband Delphi	features, user stories, requirements, use cases, function points, Web pages, GUI components, database tables, interface definitions, classes, functions/subroutines, LOC

Steve McConnell, *Software Estimation: Demystifying the Black Art*, Microsoft Press, 2006.

Estimating Effort

Informal Comparison to Past Projects – can compute an early estimate of effort (range of values) based on the effort results from analogous past projects

- Effort = PastEffort x (Size / PastSize)
- Linear model is OK, if relevant historical data is within a narrow size range



Project	Size (LOC)	Schedule (Calendar Months)	Effort (Staff Months)	Productivity (LOC/Staff Month)
Project A	33,842	8.2	21	1,612
Project B	97,614	12.5	99	986
Project C	7,444	4.7	2	3,722
Project D	54,322	11.3	40	1,358
Project E	340,343	24.0	533	639

Steve McConnell, *Software Estimation: Demystifying the Black Art*, Microsoft Press, 2006.

Estimating Effort

Formulae: If you don't have your own historical data, you can compute a rough estimate of effort based on models of industry-average efforts.

International Software Benchmarking Standards Group (ISBSG) Method

Desktop Projects:

$\text{StaffMonths} = 0.157 \times \text{FunctionPoints}^{0.591} \times \text{MaxTeamSize}^{0.810}$

Enhancement:

$\text{StaffMonths} = 0.669 \times \text{FunctionPoints}^{0.338} \times \text{MaxTeamSize}^{0.758}$

New Development:

$\text{StaffMonths} = 0.520 \times \text{FunctionPoints}^{0.385} \times \text{MaxTeamSize}^{0.866}$

Third Generation Language Project:

$\text{StaffMonths} = 0.425 \times \text{FunctionPoints}^{0.488} \times \text{MaxTeamSize}^{0.697}$

Fourth Generation Language Project:

$\text{StaffMonths} = 0.317 \times \text{FunctionPoints}^{0.472} \times \text{MaxTeamSize}^{0.784}$

Steve McConnell, *Software Estimation: Demystifying the Black Art*, Microsoft Press, 2006.

Estimating Schedule

There is a Basic Schedule Equation that produces good-enough estimates of schedule early in a medium-to-large sized project.

$$\text{ScheduleInMonths} = 3.0 \times \text{StaffMonths}^{1/3}$$

(where multiplier ranges from 2.0 to 4.0).

Steve McConnell, *Software Estimation: Demystifying the Black Art*, Microsoft Press, 2006.

Estimating Schedule

If you have historical data, you can compute a schedule estimate based on the schedules from similar past projects

Medium-to-large projects (more than 50 staff months)

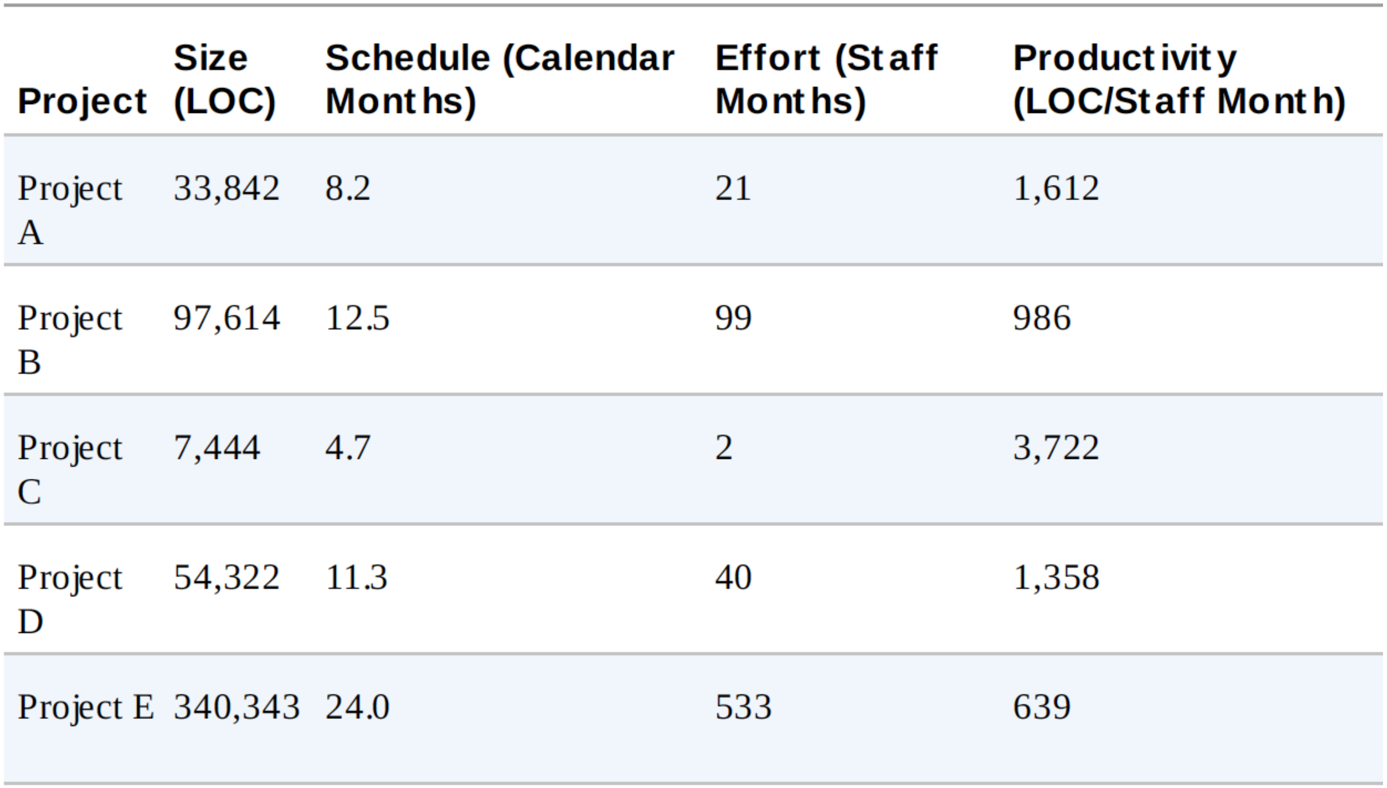
$$\text{ScheduleInMonths} = \text{PastSchedule} \times (\text{EstimatedEffort} / \text{PastEffort})^{1/3}$$

Small projects

$$\text{ScheduleInMonths} = \text{PastSchedule} \times (\text{EstimatedEffort} / \text{PastEffort})^{1/2}$$

Steve McConnell, *Software Estimation: Demystifying the Black Art*, Microsoft Press, 2006.

Estimating Schedule

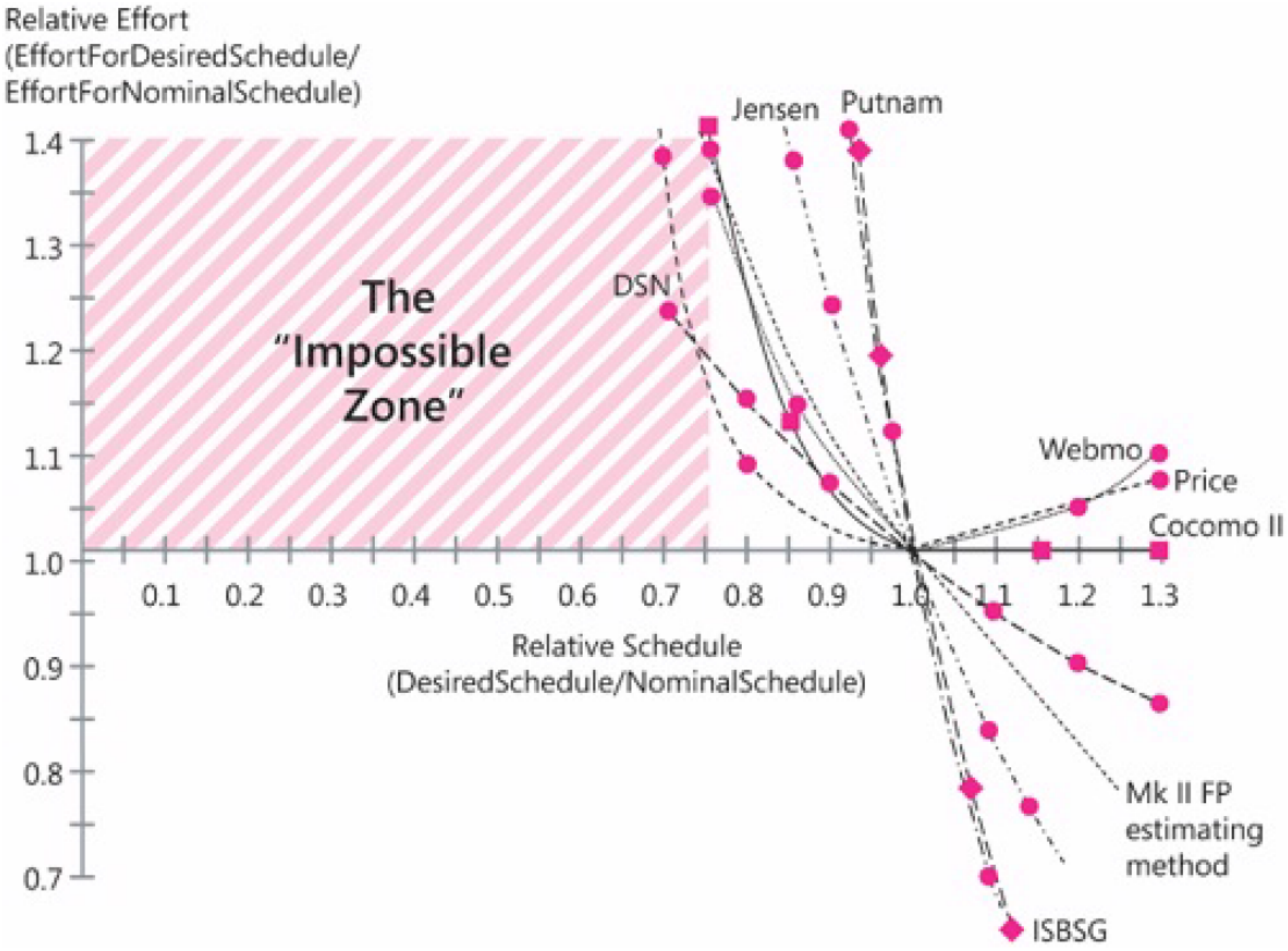


Project	Size (LOC)	Schedule (Calendar Months)	Effort (Staff Months)	Productivity (LOC/Staff Month)
Project A	33,842	8.2	21	1,612
Project B	97,614	12.5	99	986
Project C	7,444	4.7	2	3,722
Project D	54,322	11.3	40	1,358
Project E	340,343	24.0	533	639

Suppose our estimated effort is 40 to 101 staff months, with an expected effort of 65 staff months.

Steve McConnell, *Software Estimation: Demystifying the Black Art*, Microsoft Press, 2006.

Schedule Compression



Steve McConnell, *Software Estimation: Demystifying the Black Art*, Microsoft Press, 2006.

Schedule and Staffing Constraints

Schedule estimation techniques assume that, whatever the nominal schedule is, you'll be able to adjust your team size to fit the level indicated.

If staffing is fixed, then the only flexibility that you have is to reduce the feature set.

Steve McConnell, *Software Estimation: Demystifying the Black Art*, Microsoft Press, 2006.

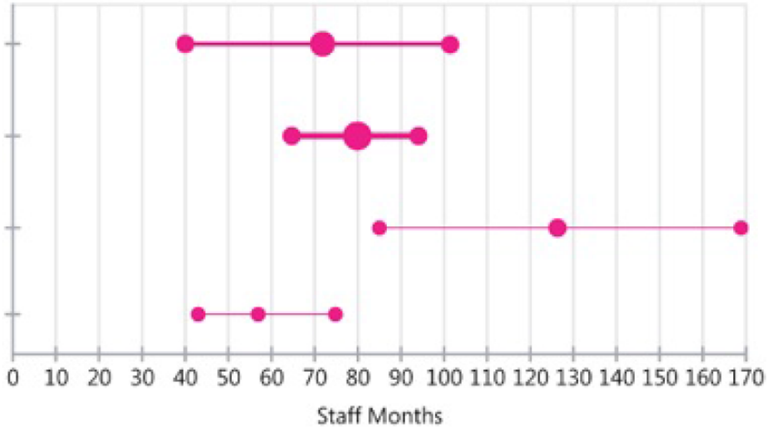
Best Practices

Decompose the estimation task into multiple subtasks and combine the results

- some estimation errors will cancel each other out

Use multiple estimation techniques and compare the results

- convergence suggests that you have good accuracy
- spread suggests some factors have been overlooked



Steve McConnell, *Software Estimation: Demystifying the Black Art*, Microsoft Press, 2006.

Best Practices

Decompose the estimation task into multiple subtasks and combine the results

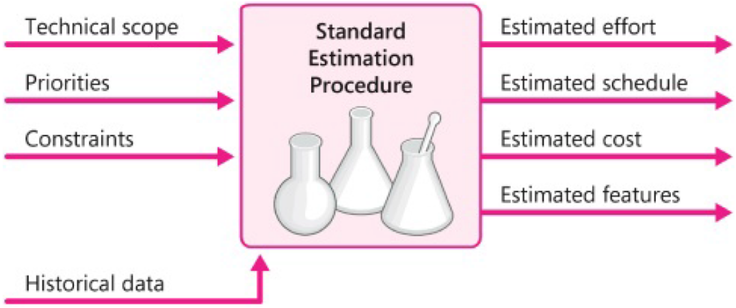
- some estimation errors will cancel each other out

Use multiple estimation techniques and compare the results

- convergence suggests that you have good accuracy
- spread suggests some factors have been overlooked

Define the estimation procedure in advance

- what granularity of task to count
- what multiple techniques to use
- when to re-estimate
- what project data to collect



Steve McConnell, *Software Estimation: Demystifying the Black Art*, Microsoft Press, 2006.

Estimates as Probability Statements

Suppose you have a software estimate for a new project or iteration

Use Case	Best Case (staff months)	Expected Case (staff months)	Worst Case (staff months)
UC 1	4.1	5.0	6.0
UC 2	5.2	6.4	7.7
UC 3	6.0	7.1	8.2
UC 4	2.4	3.1	3.6
UC 5	5.8	7.0	8.2
TOTAL	**23.5**	**28.6**	**33.7**

Steve McConnell, *Software Estimation: Demystifying the Black Art*, Microsoft Press, 2006.

Estimates as Probability Statements

Goal:

Percentage Confident	Effort Estimate (staff months)
2	23.5
10	25.4
16	26.1
20	26.5
25	26.9
30	27.3
40	28.0
50	28.6
60	29.3
70	30.0
75	30.3
80	30.8
84	31.2
90	31.8
98	33.7

Steve McConnell, *Software Estimation: Demystifying the Black Art*, Microsoft Press, 2006.

Estimates as Probability Statements

1. Combine the standard deviations of the piecemeal estimates by summing the variances of the estimates and taking the square root.

Use Case	Best Case (BC)	Worst Case (WC)	Standard Deviation (WC-BC)/6	Variance (SD^2)
UC 1	4.1	6.0	0.317	0.100
UC 2	5.2	7.7	0.417	0.174
UC 3	6.0	8.2	0.367	0.135
UC 4	2.4	3.6	0.20	0.040
UC 5	5.8	8.2	0.40	0.160
TOTAL	**23.5**	**33.7**		**0.608**
Combined Standard Deviation				**0.78**

Steve McConnell, *Software Estimation: Demystifying the Black Art*, Microsoft Press, 2006.

Estimates as Probability Statements

2. Can use a table of standard deviations to compute a percentage likelihood.

Percentage Confident	Calculation	Percentage Confident	Calculation
2	Expected Case – (2 x SD)	60	Expected Case + (0.25 x SD)
10	Expected Case – (1.28 x SD)	70	Expected Case + (0.52 x SD)
16	Expected Case – (1 x SD)	75	Expected Case + (0.67 x SD)
20	Expected Case – (0.84 x SD)	80	Expected Case + (0.84 x SD)
25	Expected Case – (0.67 x SD)	84	Expected Case + (1 x SD)
30	Expected Case – (0.52 x SD)	90	Expected Case + (1.28 x SD)
40	Expected Case – (0.25 x SD)	98	Expected Case + (2 x SD)
50	Expected Case		

Steve McConnell, *Software Estimation: Demystifying the Black Art*, Microsoft Press, 2006.

Estimates as Probability Statements

The above assumes that your piecemeal estimates have a confidence interval of 99.7% (which might be optimistic!)

More realistic estimates of the SDs of the piecemeal estimates will consider the best-case to worst-case range to have a smaller confidence interval (and thus a SD is a larger fraction of the range)

Confidence Interval	Divisor in calculations of SDs of piece estimates
10%	0.25
20%	0.51
30%	0.77
40%	1.0
50%	1.4
60%	1.7
70%	2.1
80%	2.6
90%	3.3
99.7%	6.0

Steve McConnell, *Software Estimation: Demystifying the Black Art*, Microsoft Press, 2006.

Estimates as Probability Statements

Use Case	Best Case (BC)	Worst Case (WC)	Standard Deviation (WC-BC)/2	Variance (SD^2)
UC 1	4.1	6.0	0.95	0.902
UC 2	5.2	7.7	1.25	1.56
UC 3	6.0	8.2	1.1	1.21
UC 4	2.4	3.6	0.6	0.36
UC 5	5.8	8.2	1.2	1.44
TOTAL	**23.5**	**33.7**		**5.48**
Combined Standard Deviation				**2.34**

Steve McConnell, *Software Estimation: Demystifying the Black Art*, Microsoft Press, 2006.

Estimates as Probability Statements

Percentage Confident	Effort Estimate (staff months)
25	28.6 – (0.67 x **2.34**) = ~~27.0322~~ 27
50	28.6
75	28.6 + (0.67 x **2.34**) = ~~30.1678~~ 30
90	28.6 + (1.28 x **2.34**) = ~~31.5952~~ 32

Lastly, try to report estimates in significant digits and units that are consistent with the estimate's underlying accuracy and precision

Steve McConnell, *Software Estimation: Demystifying the Black Art*, Microsoft Press, 2006.

Summary

Software estimation is critical in providing input to decisions about project scoping, planning, and tracking.

- Estimations vs. targets vs. commitments
- Techniques for estimating project size, effort, and schedule
- Sources of inaccuracies in estimations
- Data collection to support calibration in techniques
- Best practices
- Probability ranges and precision

Software Estimation for FYDP

Data Collection

In order to compute software estimates based on historical data, you need to be collecting data about your project (every 1-2 weeks)

Per requirement (e.g., use case, scenario, story, screen)

- lines of code (LOC)
- effort (staff hours)
- time (calendar hours)

Software Estimation

- effort of whole project (as sum of efforts of use cases)
- effort of next increment (as sum of efforts of atomic requirements)

Deliverable #11

Requirements-Based Estimate of Effort

Estimate the effort to develop 2 top-priority use cases, based on expert-judgment estimates of the effort to develop details about the use cases

- atomic specifications (conditions of satisfaction)
- associated UI screens

Scenario-Based Estimate of Effort

Estimate the effort to develop the rest of the use cases in your project, based on function point analysis of those use cases

Percentage Confidence in Estimates