Estimation

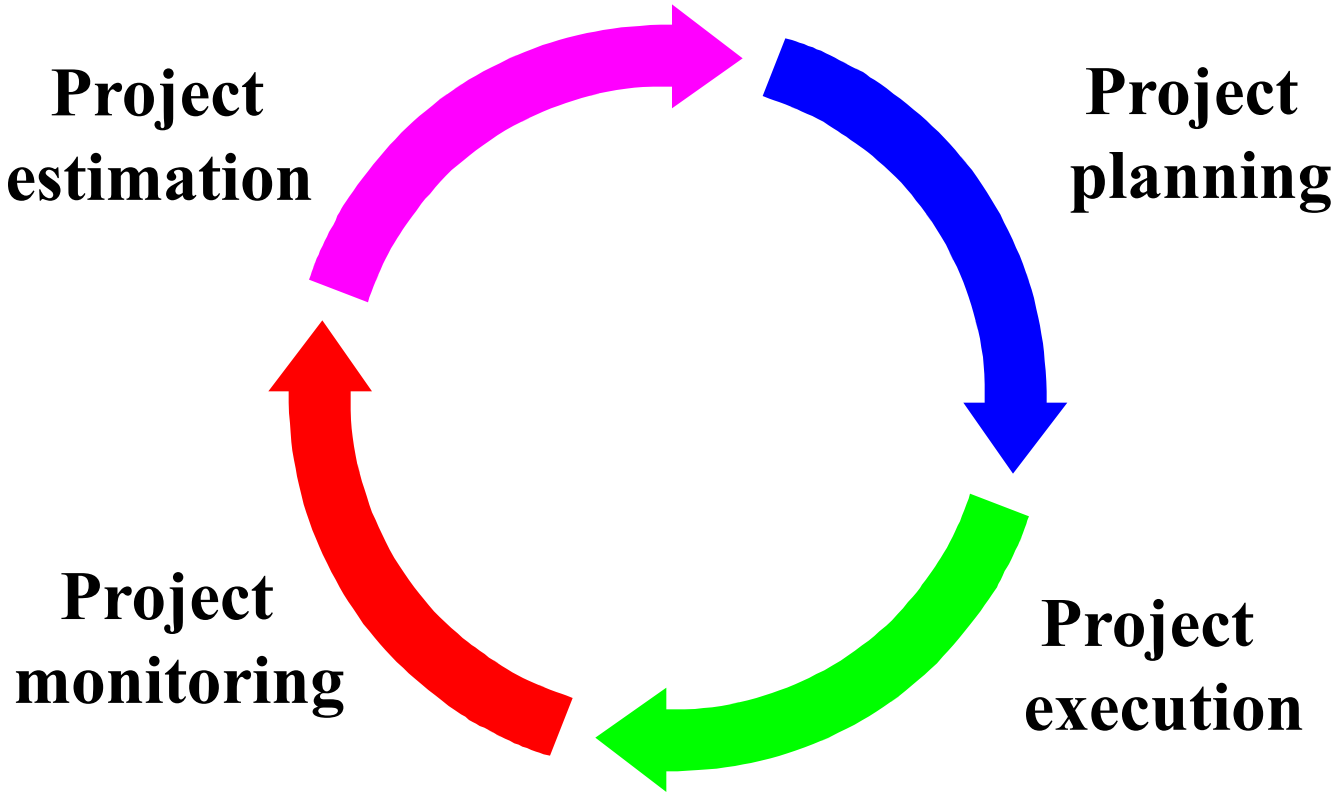
It is difficult to predict -
the future in particular.

– Do you believe in mind reading?
– Definitely, how much do you want to borrow?

After this lesson you will:

- Know a number of different estimation techniques, their advantages and disadvantages
- Be able to estimate a small project where there is much experience available
- Be able to derive an estimate from historical data
- Know about the dangers of unrealistic and ”political” estimates

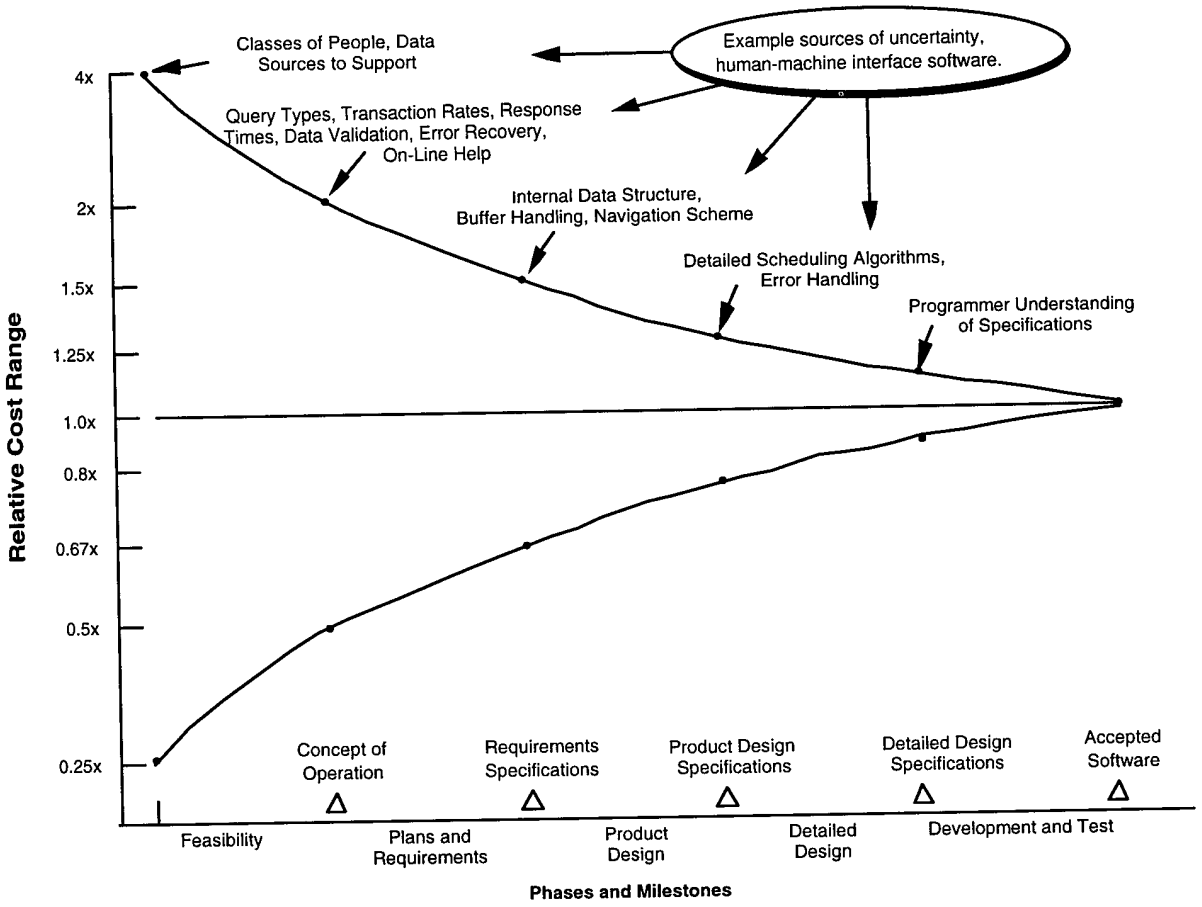
Estimation is one of the key areas of project management



To estimate is difficult

- We are not good at it
  - We are doing reasonably well when estimating minor things
  - We are doing miserably when it is about major things
- We are always too optimistic
  - Previous problems are almost always forgotten/ignored
  - Small issues are not taken into account (but they add up!)
  - Customer/management decisions may take some time (idle resources)
- Normally there is great variation in the estimating ability of individuals, depending on their knowledge and experience in the field
- The point in time on the project when the estimate is made seriously influences the estimate

Estimation accuracy



Barry Boehm, 1995

Two types of estimation techniques:

- Experience-based
  - e.g. analogies and experts, use of historic data, checklists, Successive Calculation, and DELPHI-techniques
- Model-based
  - input-parameter(s) to mathematical models, e.g. COCOMO and Function Points

Part 1
Experience-based estimation techniques

The future will be like the past, because, in the past, the future was like the past
(The axiom of experience)
- G. Weinberg, An Introduction to General Systems Thinking

Use of analogies and experts

- This activity Z resembles an activity that we previously have spent X hours on for project Y
- N.N. is an expert in this type of activity. We will ask him to make an estimate

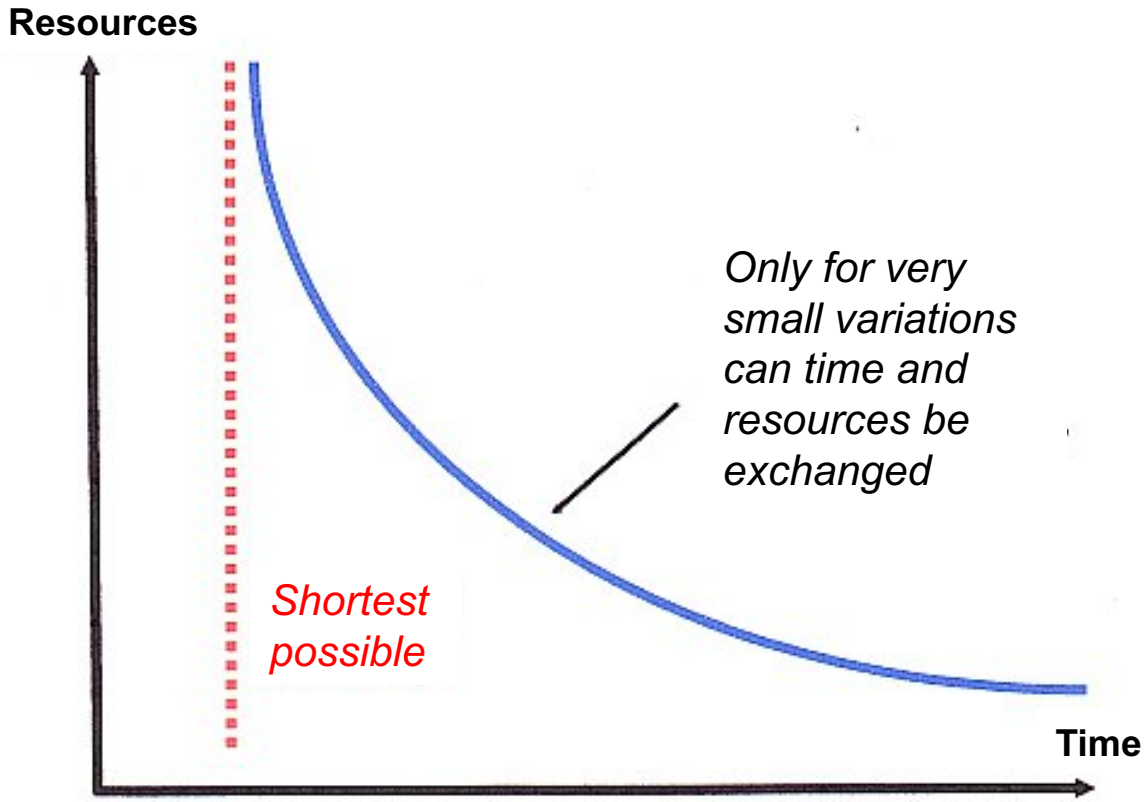
Use of historical data

- Our experience database shows, that activities of this type take X hours
  - *Example: To lay 1 m^2 tiled terrace takes 2 hours.*
- We have Y activities of that type
  - *Example: To lay 50 m^2 will take 100 hours*

Use of checklists

- Based on the knowledge/experience of the organization:
  - Standards for products and processes
    - international, company or client specific
  - Safety/Security aspects
  - Maintenance aspects
  - Educational requirements
  - Knowledge about the assignment/domain/technology
  - Etc.

Experience can only be used within limits



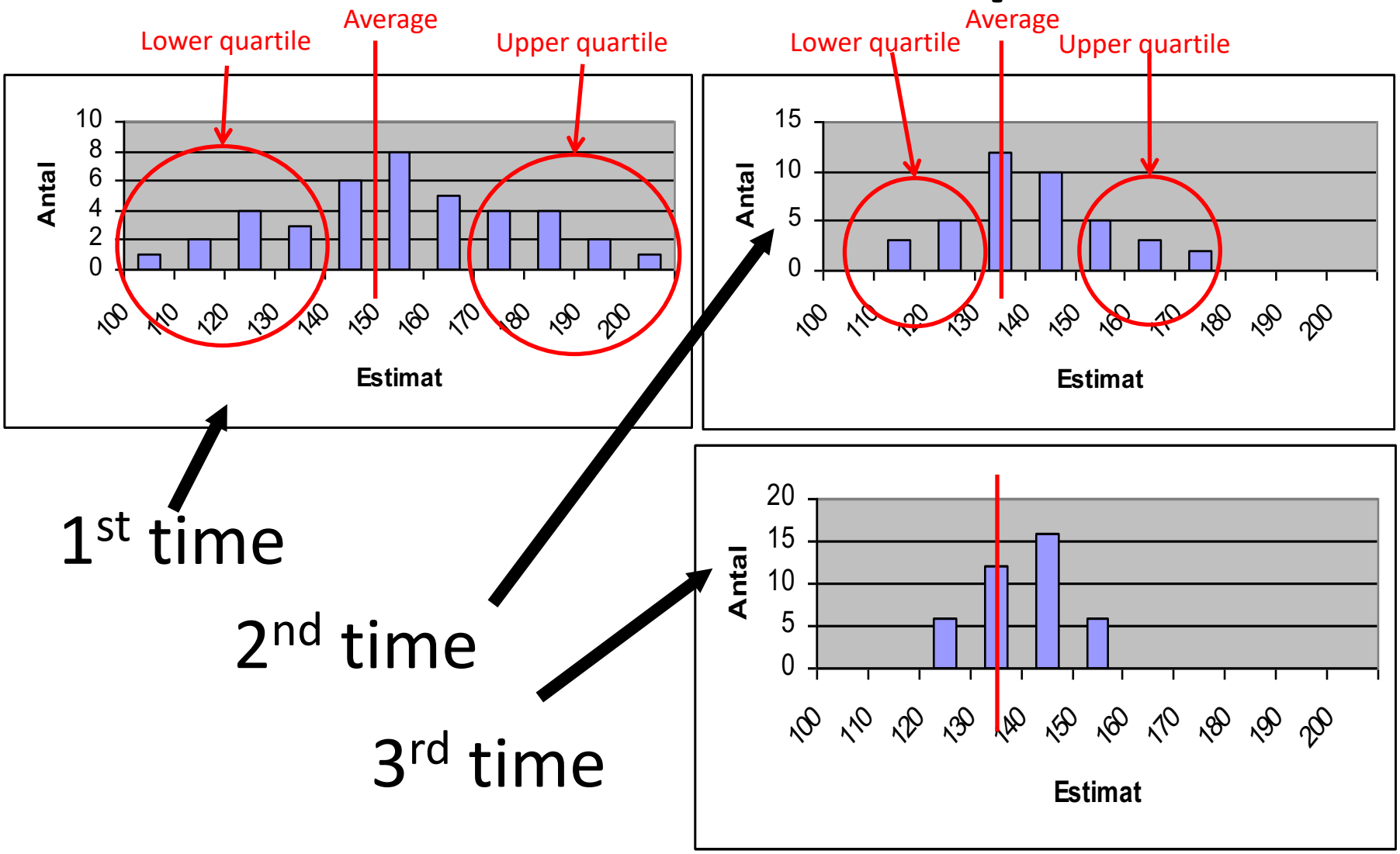
Order of experience-based estimation

- **Top-down**
  - First estimate the "whole" – the system seen as a black box - then break it down allocating **percentages** based on experience to the detailed activities. The technique can f.ex. be used for a first rough estimate very early
    - e.g. if this module is estimated at 100 hours, the coding effort will be 20%, and the …
- **Bottom-up**
  - First the "whole" is broken down into details - f.ex. in subsystems and modules. Each little part is estimated in **hours**. The sum of hours for the "whole" is calculated, possibly adjusted by percentages for certain parts based on experience
- **Bottom-up alternative**…
  - Each little part is estimated in **another amount than hours** (f.ex. number of pages, number of lines of code, number of story points). The sum of these amounts are converted to hours, f.ex. using historical data or an estimation model

DELPHI-technique

1. Each member of a group makes an estimate
2. Those with estimates in the lower and upper quartile explain to the rest of the group their reasoning behind the estimate
3. Each member of the group estimates again, now taking into consideration the arguments they have just heard
4. Step 1-3 can be repeated 2, 3, 4 or more times as needed

DELPHI-teknikken – an example



Successive Calculation – the Principle

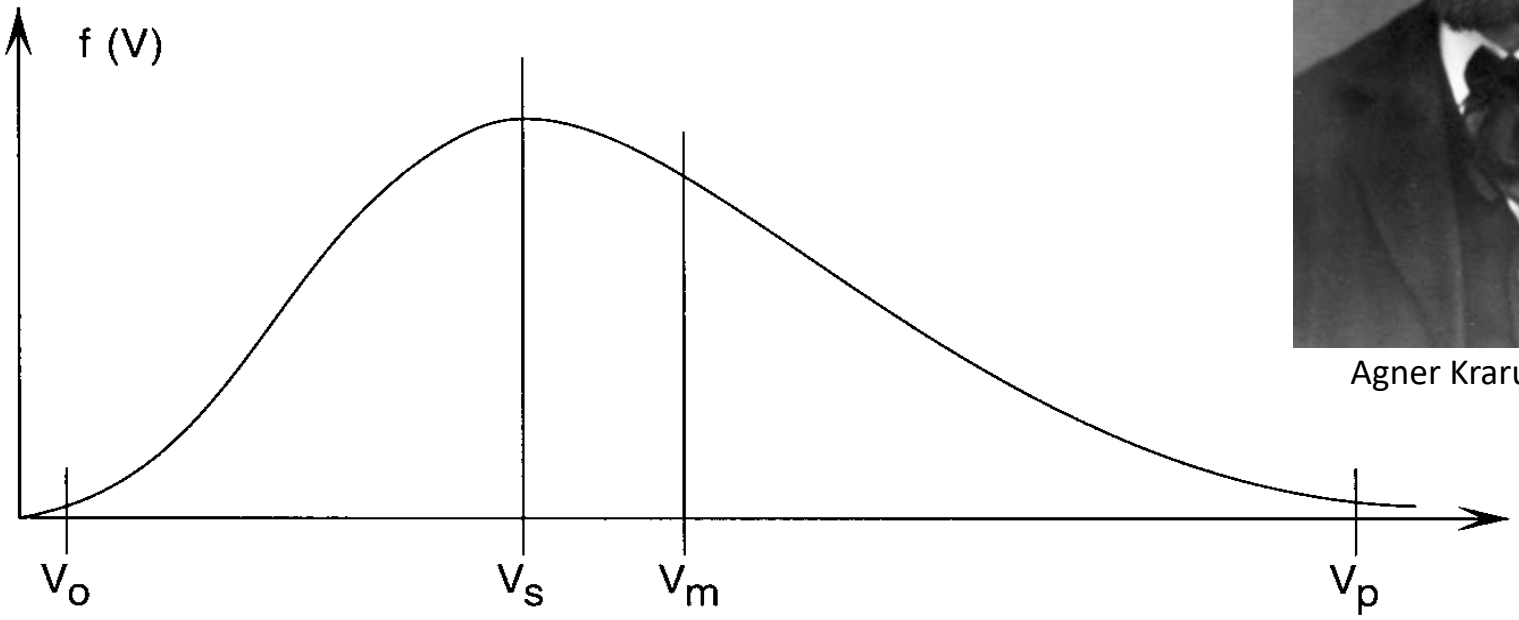
"Every pre-calculation dealing with future events … necessarily contains a number of rough estimates, which are all subject to some uncertainty"

"An old basic principle in calculation work seeks to divide the calculation into a smaller number of main items and then divide the most important of these in sub-items, which in turn - depending on the desired accuracy - are further divided."

"By ... using some basic statistical concepts I have managed to establish a method that seems to make much calculation work both more efficient and faster.“

Source: Steen Lichtenberg (1971). Rapport over successiv kalkulation. DtH. Quoted from the foreword.

Erlang distribution function (k=10)



Agner Krarup Erlang

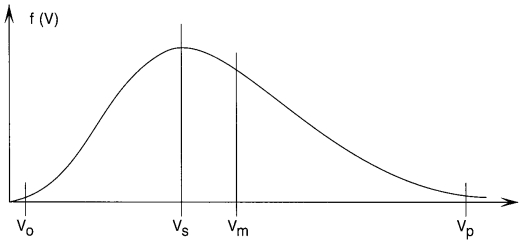
Vs = most likely value (50% fractile)

Vo = most optimistic value (1 % fractile)

Vp = most pessimistic value (99% fractile)

=> Vm = mean value (expected value)

Successive Calculation - original formulas



For an Erlang function with a k-value of 10 (where "k" is a measure for the skew of the distribution function) the **Mean value** "Vm" is calculated as:

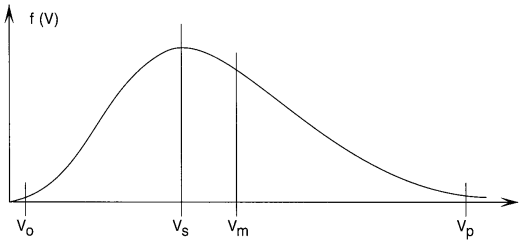
$$Vm = (Vo + 2{,}95*Vs + Vp) / 4{,}95$$

Standard deviation "S" og **Variance** "V" is calculated as:

$$S = (Vp - Vo) / 4{,}654$$
$$V = S^2$$

k = 10 is chosen "… because it is considered the most representative. For other k-values the above calculation for the mean value is subject to an error, which however for the most frequent k-values (k = 5 to 15) only amounts to a few parts per thousand." Quote: Steen Lichtenberg (1971, page 20)

Successive Calculation - formulas in practice



In real life, the following approximate formulas are used:

Vm = (Vo + 3*Vs + Vp) / 5

S = (Vp - Vo) / 5

Uncertainty can be added to the mean value so that the **probability of <u>not exceeding</u> the estimate** becomes:

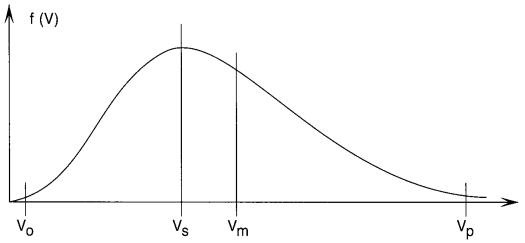
84.1% in case we use (Vm + S)

97.7% in case we use (Vm + 2S)

99.8% in case we use (Vm + 3S)

Note: Kousholt (2012e: 187) uses only probabilities in the <u>interval</u> Vm +/- nS

Successive Calculation - example



Your group has estimated an activity as:

- Vo (optimistic value) = 70 hours
- Vs (most likely value) = 80 hours
- Vp (pessimistic value) = 110 hours

Calculate Vm (mean value), S (standard deviation) and the probability limits.

When we use the approximate formulas:

- Vm = (Vo + 3*Vs + Vp) / 5 = 84 hours
- S = (Vp - Vo) / 5 = 8 hours (Note: S ~ 10% of Vm !)

- (Vm + S) = 92 hours (84.1% probability that the estimate is not exceeded)
- (Vm + 2S) = 100 hours (97.7% probability that the estimate is not exceeded)
- (Vm + 3S) = 108 hours (99.8% probability that the estimate is not exceeded)

Successive Calculation – the procedure

The idea is to dig deeper into the uncertain elements (largest S values).

The method points out which activities are in need of being broken further down in order to reduce the uncertainty of the estimate.

The whole procedure:

1. Break down what must be estimated into main activities
2. For each activity give an optimistic, a pessimistic and a most likely value (Vo, Vp, Vs)
3. Mean value (Vm), Standard deviation (S), and Variance ($V=S^2$) is calculated for each activity
4. The activity with the largest standard deviation (S) has the highest influence on the total certainty of the calculation. This activity is then broken down into sub-activities, which together replace the original activity
5. Point 2 and 3 are repeated for each sub-activity
6. Point 2-5 is repeated until the total standard deviation is acceptable

see formulas and example in Kousholt (2012e: 185-187)

Calculating the total Standard deviation

- In order to find the **total** Standard deviation for a project that has been broken down into several (sub)activities you must use the Variance
- The **Variance (V)** for each (sub)activity equals the square of the Standard deviation (S):

$$\mathbf{V = S^2}$$

- The **total** Standard deviation S_{total} for the project is then equal to the square root of the sum of variances for each of the project's activities:

$$S_{total} = \sqrt{\left(\sum_{j=1,n} \left(V_j \right) \right)}$$

Caution!

- Standard deviation calculated as (on previous slide):

$$S_{total} = \sqrt{\left(\sum_{j=1,n} \left(V_j \right) \right)}$$

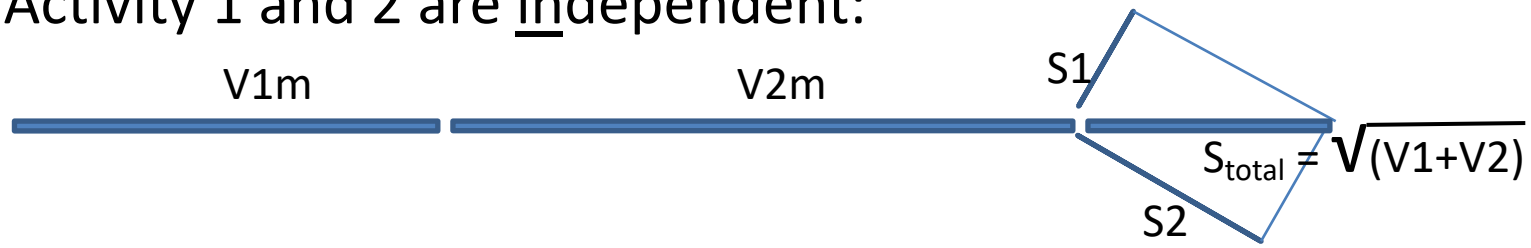
  requires the estimates of the individual activities to be **independent**, i.e. that an overrun on one does not affect the estimate of others.

- Standard deviation for **dependent** activities must be calculated as the direct sum:

$$S_{total} = \sum_{j=1,n} \left(S_j \right)$$

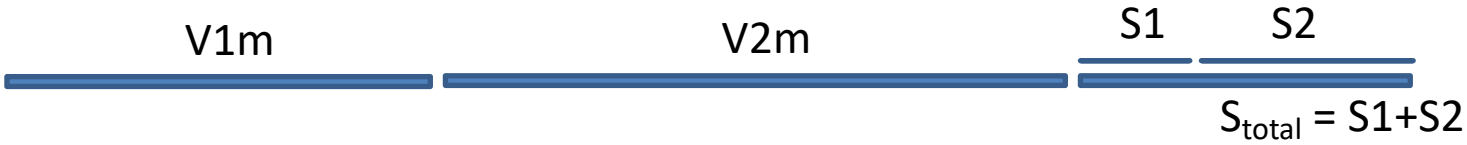
Independence vs. Dependence - illustrated

Activity 1 and 2 are independent:



where $Vj=Sj^2$

Activity 1 and 2 are dependent:



Successive Calculation example (1:3)

	Vo	Vs	Vp	Vm	S		V
Alpha	20	25	35	26	3		9
Bravo	60	85	115	86	11		121
Charlie	55	65	100	70	9		81
Delta	15	20	30	21	3		9
Echo	35	50	65	50	6		36
Total				253		ΣV=	256
					√ ΣV=	16	

Successive Calculation example (2:3)

The activity Bravo is broken down into sub-activities, because it has the largest Standard deviation

	Vo	Vs	Vp	Vm	S		V
Alpha	20	25	35	26	3		9
Bravo-1	20	25	40	27	4		16
Bravo-2	30	35	45	36	3		9
Bravo-3	15	25	30	24	3		9
Charlie	55	65	100	70	9		81
Delta	15	20	30	21	3		9
Echo	35	50	65	50	6		36
Total				254		ΣV=	169
					√ΣV= 13		

Successive Calculation example (3:3)

The activity Charlie is then broken down into sub-activities, as it now has the largest Standard deviation

	Vo	Vs	Vp	Vm	S		V
Alpha	20	25	35	26	3		9
Bravo-1	20	25	40	27	4		16
Bravo-2	30	35	45	36	3		9
Bravo-3	15	25	30	24	3		9
Charlie-1	25	35	60	38	7		49
Charlie -2	20	20	30	22	2		4
Charlie -3	6,33	11,22	15	11	1,73		2,99
Delta	15	20	30	21	3		9
Echo	35	50	65	50	6		36
Total				255		ΣV=	144
					√ ΣV=	12	

Continue to break down until ...

Rule of thumb from Lichtenberg (1971, page 14):

"The item or items that have the greatest variance ... divided into a number of sub-items ... continues until you either reach a satisfactory accuracy or meet uncertainties that can not be reduced by splitting or otherwise."

Rule of thumb from Peter Willkan, A.P. Møller, at a Workshop on estimation in Dansk Dataforening:

"Continue to break down activities until the standard deviation is less than one-twentieth of the mean e.g. S < Vm/20"

However, experience shows that it is often unrealistic to reach such small uncertainties as one twentieth (5%).

Managing general uncertainty

	Vo	Vs	Vp	Vm	S	V
Alpha	20	25	35	26	3	9
Bravo	60	85	115	86	11	121
Charlie	55	65	100	70	9	81
Delta	15	20	30	21	3	9
Echo	35	50	65	50	6	36
Uncertainty about CASE-tool	-50	0	100	10	30	900
Total				263	ΣV=	1156
				√ΣV=	34	

Exercise 1: Successive Calculation

The exercise is about making a fruit salad as described in:

- 82a_Exercise in successive calculation

You can find the Successive Calculation formulas in:

- 82b_Successive calculation formulas.xlsx

- Form groups of 3 persons
- Spend app. 10 minutes individually estimating each of the activities
  - e.g. Vo, Vp, Vs **and** calculate Vm and S
- Spend app. 15 minutes in the group for making a common estimate

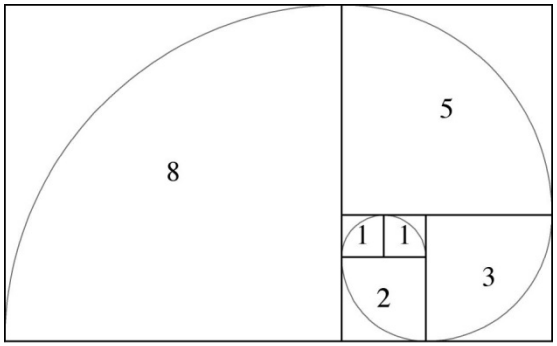
Summary in the plenary

Estimation for iterative models

- The requirements are expressed in the form of short scenarios (user stories)
- The idea is that it is easier to assess the relative (greater than, more complex than) than the absolute (40 hours, 3 man-weeks)
- Used fex. in SCRUM

- Requires experience from previous stories, which can be compared
- Works best when there are many stories to compare
  - Also enables a prioritization/selection in which iteration/sprint they should be developed

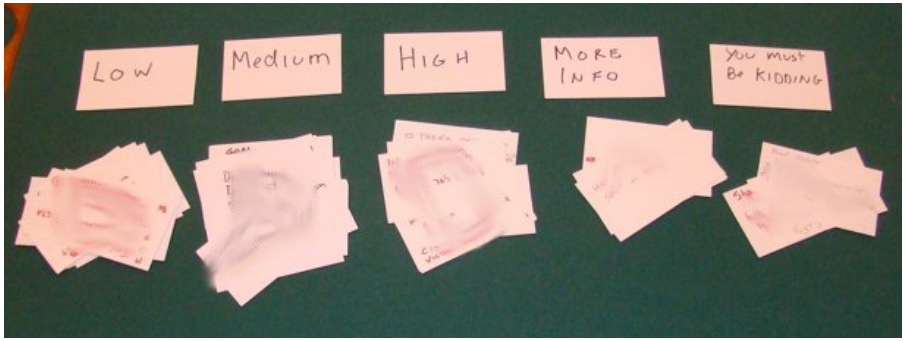
Planning Poker



- Variant of the Delphi-technique
- Uses Fibonacci numbers (1, 2, 3, 5, 8, 13, 21 ...)
- Normally the following are added:
  - ∞ (too big for one iteration, must be broken down)
  - ½ (for stories, that require no real effort)
  - ? (don't know, requires more information/detail)
- The reference (1 story point) is agreed upon, i.e. a previously completed very simple story is chosen
- The assumption is, that a story of 3 points takes 3 times longer than a story of 1 point
- Each person presents his estimate for the task ("plays his card")
- The two most divergent (highest and lowest) each explain their estimate
- Another game is played and results discussed until agreement is reached
- The estimate is converted to concrete effort (e.g. hours) using the reference

J.W. Grenning: www.renaissancesoftware.net

Affinity estimation



- Variant of the Delphi-technique
- Uses the categories:
  - easy, medium, difficult,
  - unknown (i.e. more information)
- All stories (written on index cards) are put on the table/blackboard
- A card is read aloud, discussed/explained and placed in a category
  - pick those first, which seem to appear smallest/easiest
  - put it in ”unknown” if agreement cannot be reached immediately

- Then the cards are put in order from smallest to largest (piled when the same) and assigned a factor relative to the smallest – start with those in the category ”easy”
- The smallest story is converted to concrete effort (e.g. hours) in relation to previously completed stories – the rest are then calculated from this

J.W. Grenning: www.renaissancesoftware.net

Advantages and disadvantages of methods and techniques

Method	Advantages	Disadvantages
Analogies and Experts	• Based on typical experiences • Takes exceptions into account	• No better than the experience and knowledge of the "experts" • Often "Biased" (too optimistic or pessimistic)
Historical data	• Representing concrete experience • Comprehensive data	• Seldom matches precisely, because data are not in context • For IT: Based on lines of code
Successive Calculation	• Gives better discussions • Uncertainty can be calculated	• A lot of number-juggling • Garbage in – garbage out
DELPHI-techniques	• Good use of experts	• Requires previous experiences • Not good for many details
Top - down	• Focus on the whole • Fast and effective • Can be used early in the project	• Based on very few details • Often subject to great uncertainty
Bottom - up	• Based on details • More stable • Promotes individual "commitment"	• Some costs may be forgotten because you cannot "see the forest because of the trees". • Requires many resources

Part 2
Model-based techniques

Estimation based on input parameter(s) to mathematical models with accompanying adjustment factors

Examples:

- COCOMO
- Function points

COCOMO

- COnstructive COst MOdel
- An estimation technique originally published by Barry Boehm in:
  - Barry Boehm (1981), *Software Engineering Economics*, Prentice Hall
- Updated several times e.g. in 1995 to COCOMO 2.0
  - Barry Boehm (2000), *Software Cost Estimation with COCOMO II*, Prentice Hall
- Based on estimates of lines of source code
  - But how do we get from a requirement specification to the number of lines of source code ?

COCOMO: ”Organic” development

Relatively small teams develop software of a well-known type for in-house use. The majority of the group has experience with similar systems in the organization.

$$\mathbf{PM = 2.4\ (KDSI)^{1.05}}$$

$$\mathbf{MD = 2.5\ (PM)^{0.38}}$$

PM = Person Months KDSI = Thousand lines of source code

MD = Months of development

COCOMO example: ”Organic”

Example: We are going to develop a program estimated at 10,000 lines of source code, and the type of development is ”organic”.

$PM = 2.4\ (10)^{1.05} = 27$ person months

$MD = 2.5\ (27)^{0.38} = 8.7$ months of development

Average man-power: 27 / 8.7 = **3.1 persons**

COCOMO: Embedded

Embedded development

The project may consist of new technology, algorithms we don't know well, or a new innovative development method. The most characteristic is, that the project is subject to many (tight) constraints

$PM = 3.6\ (KDSI)^{1.20}$

$MD = 2.5\ (PM)^{0.32}$

Example with 10,000 lines of code (10 KDSI):

$PM = 3.0\ (10)^{1.20}$ = 57 person months

$MD = 2.5\ (57)^{0.32}$ = 9.1 months of development

Average man-power: 57 / 9.1 = 6.3 persons

COCOMO: ”Semi-detached”

”Semi-detached” development

In between ”organic” and ”embedded”

$PM = 3.0\ (KDSI)^{1.12}$

$MD = 2.5\ (PM)^{0.35}$

Example with 10,000 lines of code (10 KDSI):

$PM = 3.0\ (10)^{1.12}$ = 40 person months

$MD = 2.5\ (40)^{0.35}$ = 9.1 months of development

Average man-power: 40 / 9.1 = 4.4 persons

Extended COCOMO (1:2)

PM = 3.0 (KDSI)$^{1.12}$ * f1 * f2 * f3 … f14 * f15 (semi detached)

SOFTWARE FACTORS

f1 = Reliability required of the system (RELY)

f2 = Size of database (DATA)

f3 = Complexity of the system (CPLX)

COMPUTER FACTORS

f4 = Requirements for speed in operations (TIME)

f5 = Requirements for main storage in operations (STOR)

f6 = Frequency of changes to the technical platform (VIRT)

f7 = Wait times for results (TURN)

Extended COCOMO (2:2)

PEOPLE FACTORS

f8 = Capability / Competences of the analysts (ACAP)

f9 = Developers' experience with the application domain (AEXP)

f10 = Capabilities / Competences of the programmers (PCAP)

f11 = Developers' experience with the technical platform (VEXP)

f12 = Programmers' experience with the programming language (LEXP)

PROJECT FACTORS

f13 = The degree of use of modern programming methods (MODP)

f14 = Use of programming tools (TOOL)

f15 = Requirements on schedule / delivery time (SCED)

COCOMO 2.0 (1995)

NEW FACTORS IN COCOMO 2.0

n1 = Requirements for documentation

n2 = Degree of required reuse

n3 = Developer continuity

n4 = Development spread over more locations

n5 = Organizational experience with the project type

n6 = Flexibility of requirements

n7 = Degree of clarification of the assignment

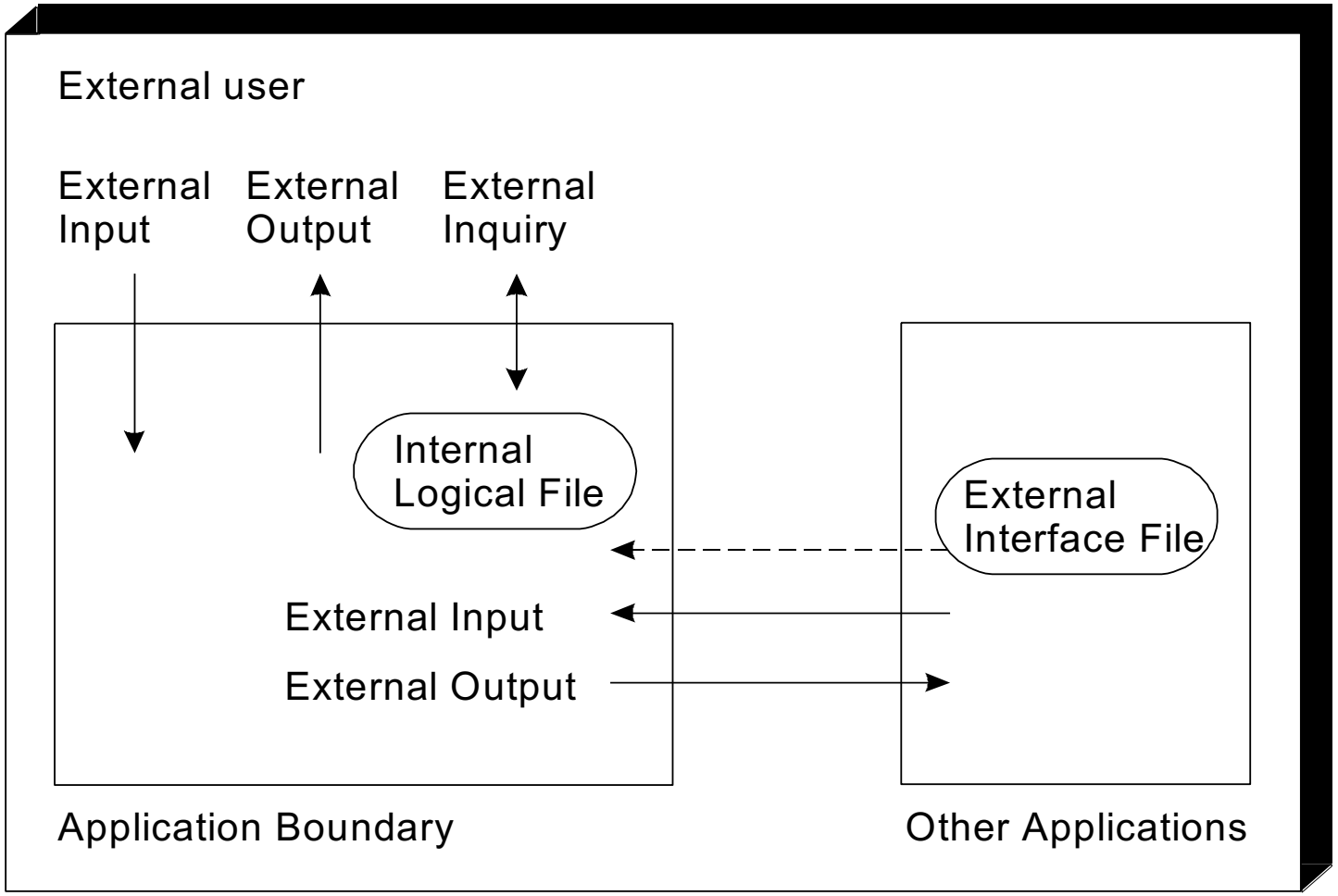
n8 = Team spirit

n9 = Professional maturity of the development organization

Function Points

- A factor-based estimation technique originally published by Albrecht (1979)
- Revised 1984, 1986, 1990 … etc.
- SPR - Software Productivity Research defines Feature Points in 1986
- IFPUG - International Function Points User Group. Permanent in 1992. Version 4.0 in 1994.
- The standard methodology to count function points is an ISO standard (ISO/IEC 20926) and supported by the International Function Point User Group (IFPUG.)

Five things are counted



Example:
Function Point calculator

FUNCTION POINT CALCULATOR

APPLICATION NAME _____

FUNCTION POINT FACTORS	RAW COUNT	EMPIRICAL WEIGHT	UNADJUSTED TOTALS
NUMBER OF INPUTS?	_____	× 4 =	_____
NUMBER OF OUTPUTS?	_____	× 5 =	_____
NUMBER OF INQUIRIES?	_____	× 4 =	_____
NUMBER OF DATA FILES?	_____	× 10 =	_____
NUMBER OF INTERFACES?	_____	× 7 =	_____
UNADJUSTED TOTAL			_____
COMPLEXITY MULTIPLIER (0.65 TO 1.35)			_____
FINAL ADJUSTED FUNCTION POINT TOTAL			_____
PRINT RESULTS (Y/N)?	_____		
SAVE FILE TO DISK (Y/N)?	_____		
RUN PROGRAM AGAIN (Y/N)?	_____		

Example of Function Point counting

A screen with external input =1 (EI). A screen with output data and possibility for printing a copy of the screen = 2 external output data (EO). Internal files =1 (ILF).

Parameter	Number		Weight		Total
EI - External input data	1	x	4	=	4
EO - External output data	2	x	5	=	10
EQ - External queries	0	x	4	=	0
ILF - Internal logical files	1	x	10	=	10
ELF - External interfaces	0	x	7	=	0
Non-adjusted sum					24
Adjustment multiplier					0,75
Adjusted sum					18

Function Point adjustment

Adjustment can be performed according to several models ranging from +/- 25% to very advanced models ranging up to +/- 125%

In the example IBM´s 1984 method is used (until 1995 corresponding to IFPUG´s approach)

Adjustment multiplier
= (SUM*0,01) + 0,65
= (10*0,01) + 0,65 = 0,75

24 non-adjusted FP * 0,75
= 18 adjusted Function Points

Data Communications	0
Distributed functions	0
Heavily used configuration	0
Transaction rate	0
On-line data entry	2
End-user efficiency	3
On-line update	2
Complex processing	0
Installation ease	0
Operational ease	3
Multiple sites	0
Facilitated change	0
TOTAL INFLUENCE SUM	10

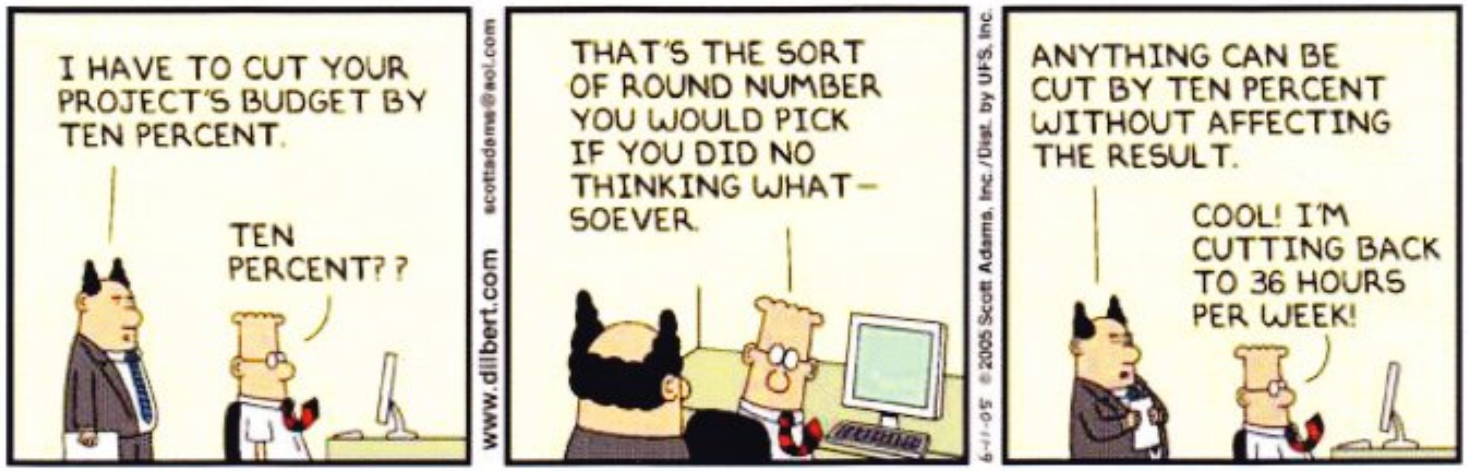
Supplementary literature on model-based estimation

- Hughes & Cotterell (2006), *Software Project management*, McGrawHill, Chapter 5 – Software effort estimation
- IFPUG Counting Practices Committee (1990), *Function Point Counting Practices Manual: Release 3.0*
- Kitchenham B.A. (1991), *Empirical Studies of the Assumptions Underlying Software Cost Estimation Models*, NCC Ltd.
- Symons C.R. (1991), *Software Sizing and Estimating: Mk II FPA,* John Wiley & Sons

Advantages and disadvantages of model-based methods and techniques

Method	Advantages	Disadvantages
COCOMO	• Known by many; part of many books • Several levels of use	• Isn't used much • Data from large, US projects -> OK in Denmark?
Function Points	• Very fine precision • Many experiences of use (also in DK)	• Takes a long time to learn • Certification of counters • Huge investment before you have your own data

Wrapping up
– some important remarks



Estimates and Uncertainty

- Estimates are predictions about the future. Predictions are subject to uncertainty
- Always include the uncertainty in every estimate
  - Example: "My estimate is that it will take between 2000 and 2800 hours to do the job"
- List all the preconditions/assumptions you have made for the estimate
- If you must give one number then use the median (in stead of the average) e.g. the prediction which has the same probability of being above and below

Buffer – the project manager's reserve

- Shall cover expected costs, where at estimation time it is not possible to ascertain where, when, or how much. One sum is allocated to cover such special situations
  - Can f.ex. be +1*S or +2*S as calculated through Successive Calculation
- Released by the project manager for those activities that during the execution lie over the mean value to cover the deficit
- Reduced as the project progresses

”Contingency” – the budget reserve

- Shall cover expected costs, where at estimation time it is not possible to ascertain where, when, or how much. One lump sum is allocated to cover such special situations
- ”Contingency” is also called: budget reserve, management reserve, sundries, or unforeseen costs
- ”Contingency” is usually defined as a percentage of the other cost items. The percentage varies with line of business and project type
- Released by the Steering group when needed

Estimates are never to be reduced by political conditions !!

- Do not let yourself be pressed by political conditions – Nobody wins in the long run. Estimates are not subject to politics – the release date of the project often.
- If neither resources or calendar time are sufficient, management or the Steering group must reduce the scope of the project (remember the ”steel triangle”)
- Remember to document the decisions!

Political estimation: Some NOT recommendable methods

- **Parkinson.** The work expands to cover the available working time.
- **Win the contract**. The estimate is fixed so you can win the contract.
- **To fit the deadline**. The estimate is retrofitted based on when we must deliver with the resources allocated.
- **As much as allowed**. The estimate is equal to the previous estimate + the delay which the client/boss is willing to accept. Or the estimate is equal to the "right" (expected) answer – often defined externally
- **Multiply by π + 10%**. This and similar formulas can of course reflect the uncertainty of a project. But how do we know that it does ?