Software Architecture and Engineering

Project Cost Management

Peter Müller

Chair of Programming Methodology

Spring Semester 2012

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

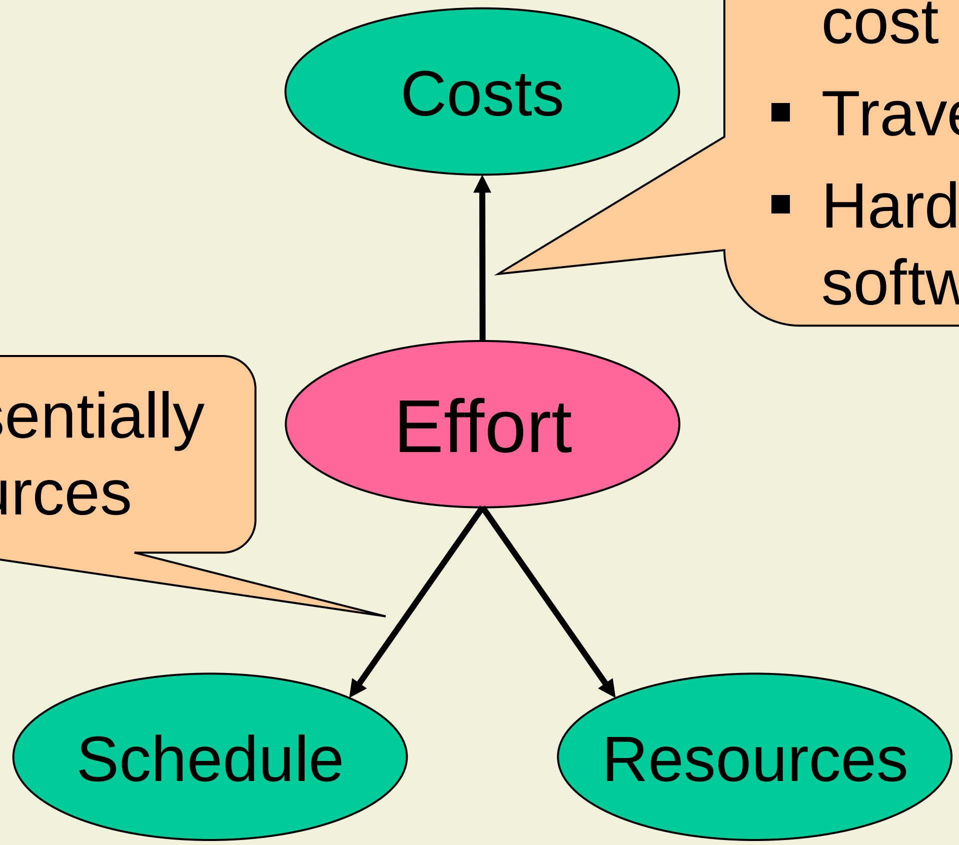
12. Project Cost Management

12.1 Software Estimation

12.2 Budgeting

12.3 Earned Value Method

Estimations in Software Projects



- Mostly personnel cost (effort)
- Travel, training
- Hardware, software

Duration is essentially effort / resources

Estimation Exercise

- How many passenger planes does Lufthansa have?
  - Not counting subsidiaries
- How can we approach this problem systematically?

Empirical Estimation: Expert Judgment

- Estimate is based on **experience and historical data**
- Involve experts in
  - Development techniques
  - Application domain
- Most **common technique** in practice

Top-Down Estimation

- Estimation by **analogy**
  - Comparison with **similar projects**
  - Analysis of differences
  - Typical example: SAP introduction

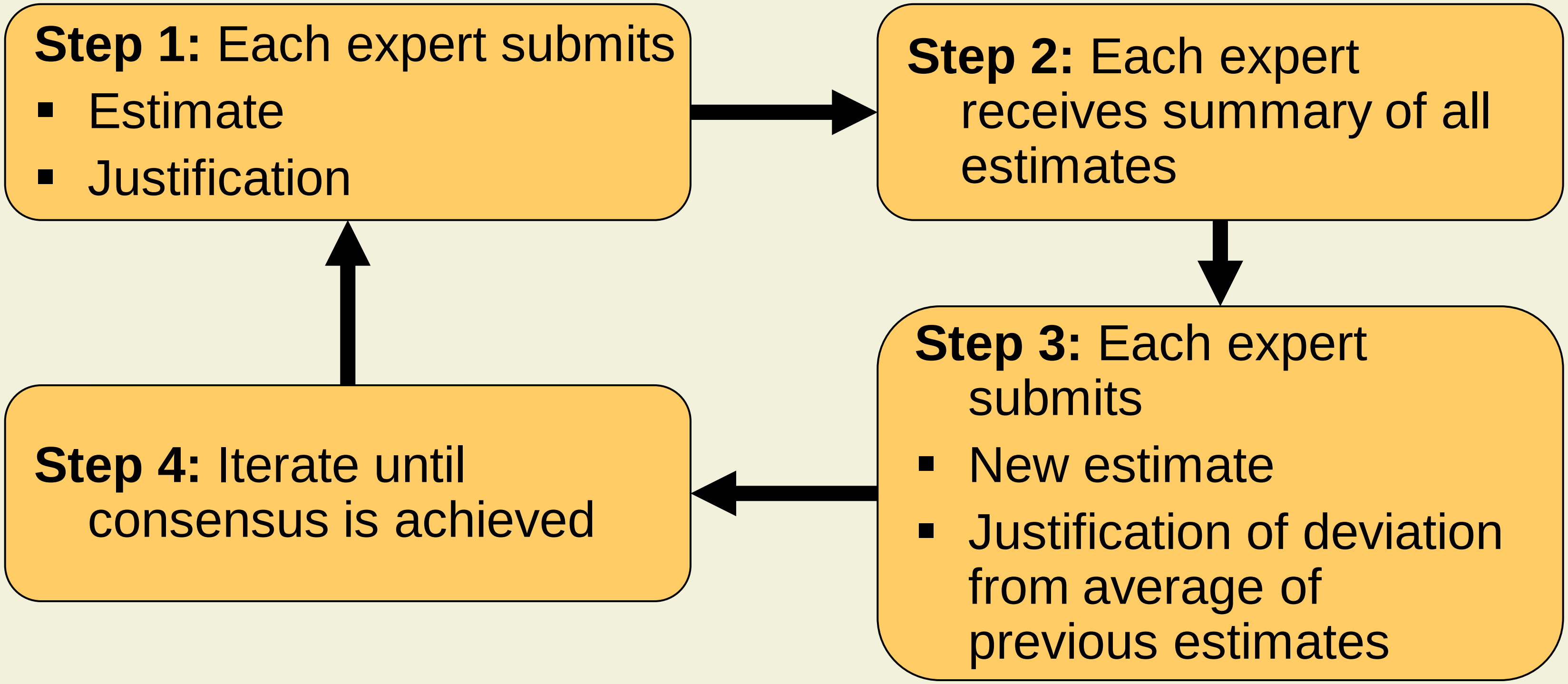
Pros

- Quicker and less expensive than other methods
- Can be done early in the project

Cons

- Underestimation of difficult technical problems likely
- No detailed justification of estimate
- Be aware of scalability problems!

Top-Down Estimation: Delphi Method



Step 1: Each expert submits
- Estimate
- Justification

Step 2: Each expert receives summary of all estimates

Step 3: Each expert submits
- New estimate
- Justification of deviation from average of previous estimates

Step 4: Iterate until consensus is achieved

- **More accurate** than ordinary expert judgment
  - Eliminates outliers
- **More expensive** to produce

Bottom-Up Estimation

- Estimation by **decomposition**
    - Estimating the effort for **individual work packages**
    - Cost and accuracy depend on size of the work packages

Pros

- See “cons” of top-down estimation

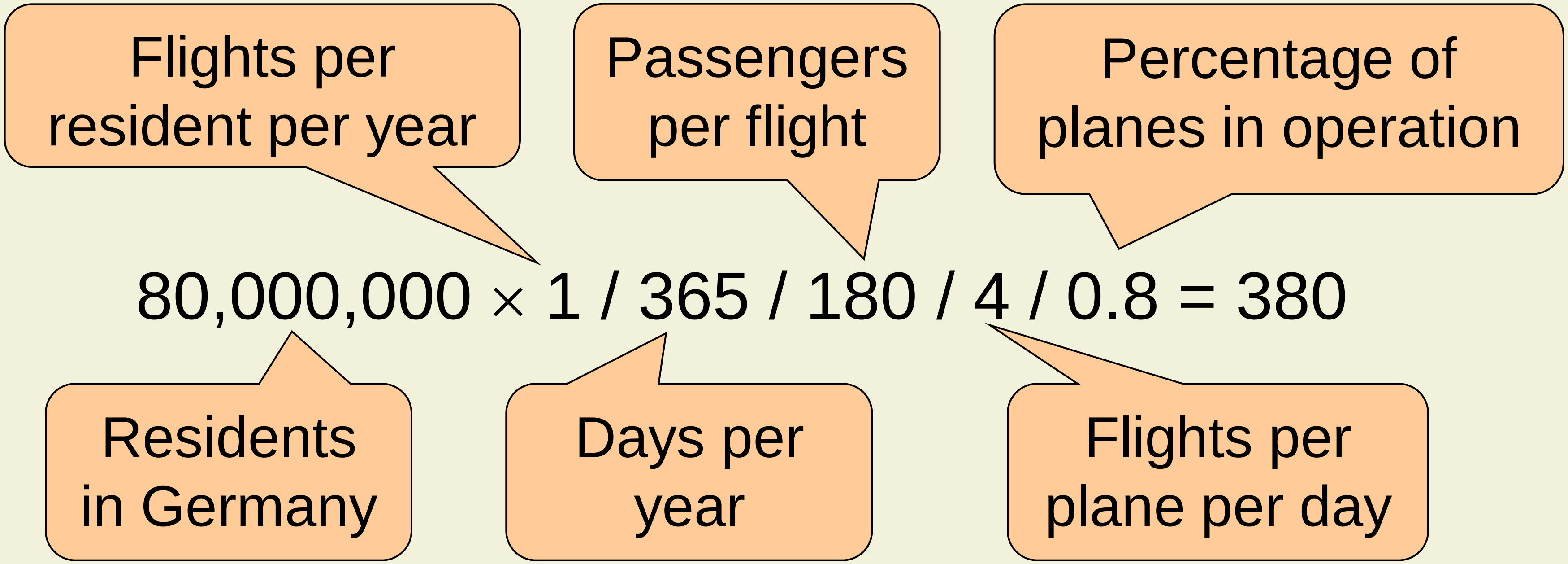
Cons

- Underestimation because effort does not grow linearly (due to complexity, etc.)
- Underestimation of integration effort
- Requires initial system design

Program Evaluation and Review Technique

- Goal: Manage probabilities with simple statistics
- Approach: Ask several experts for three estimates
  - **Optimistic, Likely (mode), and Pessimistic**
- Important formulas
  - Mean $M = (O + 4 \times L + P) / 6$
  - Deviation $V = (P – O) / 6$
- Assumptions
  - Project effort is normally distributed (more than 20 work packages)
  - Work package efforts are statistically independent (ignores single underlying cause of delay)

Algorithmic Estimation



Flights per resident per year

Passengers per flight

Percentage of planes in operation

$$80{,}000{,}000 \times 1 / 365 / 180 / 4 / 0.8 = 380$$

Residents in Germany

Days per year

Flights per plane per day

- Algorithmic estimation is based on
  - **Cost model**, represented by formula
  - **Measurement of size** (passengers, destinations, etc.)
  - **Parameters** (size of planes, planes in operation, etc.)

Algorithmic Estimation of Software

- Basic cost model

$$\boxed{\text{Effort} = A \times \text{Size}^{B} \times m(X)}$$

- Size: Some measurement of the software size
- A: Constant factor that depends on
  - Organizational practices
  - Type of software
- B: Usually lies between 1 and 1.5
- X: Vector of cost factors
- m: Adjustment multiplier

Cost Models

$$\boxed{\text{Effort} = A \times \text{Size}^B \times m(X)}$$

- Cost models
  - Define a way to determine the size
  - Define cost factors X
  - Provide defaults for parameters A, B, m (based on hundreds of projects)

- Important examples
  - Function point analysis
  - Constructive cost model (COCOMO)

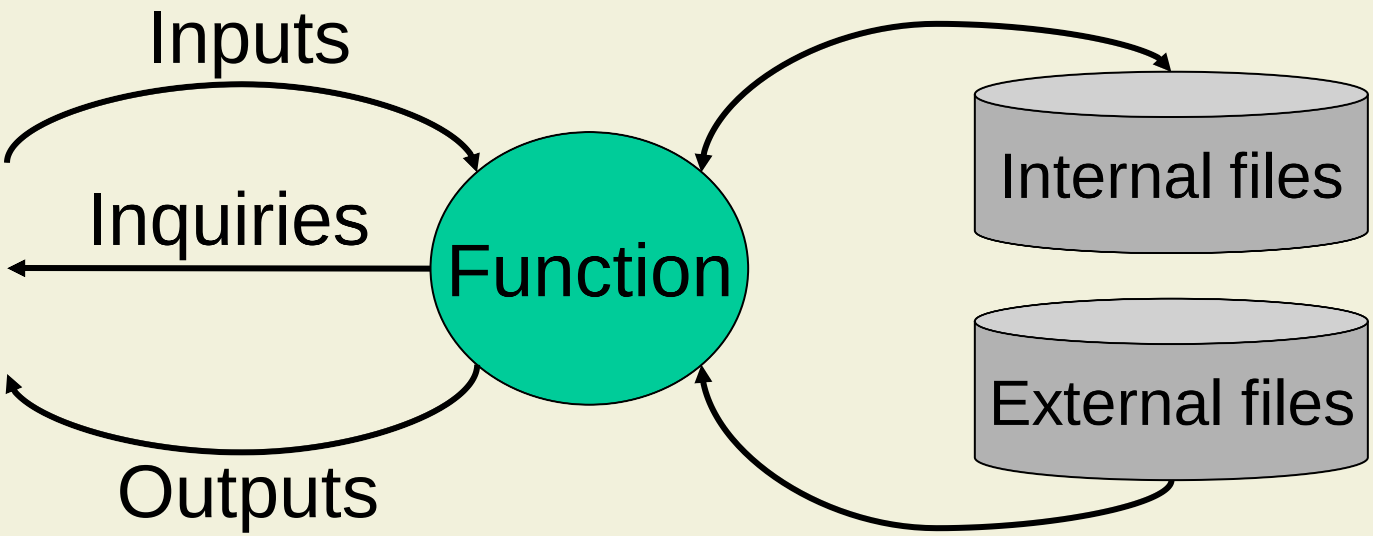
Measuring Size: Lines of Code

- Software size can be measured in lines of source code
  - Most commonly used metric

- **Difficult in early phases** of the project (before design is known)
  - Reuse, make-or-buy decisions
- **Influenced** heavily by choice of **programming language**
- Should only be **used indirectly**

Function Point Analysis

- Size is estimated based on **requirements**



Functions

- **Inputs**
  - Forms, dialogs, messages, XML documents
- **Outputs**
  - Web pages, reports, graphs, messages, XML documents
- **Inquiries** (input/output combinations)
  - Simple web inputs, generally producing a single output
- **Logical internal files** (controlled by the program)
  - Tables, views or files in database
- **External files** (controlled by other programs)
  - Tables or files used from other systems or databases

Complexity of Functions

- Determine **complexity** of each function

Input	Simple	Average	Complex
Data elements	1-5	6-10	>10
Checking	Formal	Formal, logical	Formal, logical, requires DB access

- **Weight** each function according to complexity

Factor	Simple	Average	Complex
Inputs	3	4	6
Outputs	4	5	7
Inquiries	3	4	6
Ext. files	7	10	15
Int. files	5	7	10

Cost Factors

- Data communications
- Distributed processing
- Performance
- Heavy use
- Transaction rate
- Online data entry
- Complex interface
- Online data update
- Complex processing
- Reusability
- Installation ease
- Operational ease
- Multiple sites
- Facilitate change

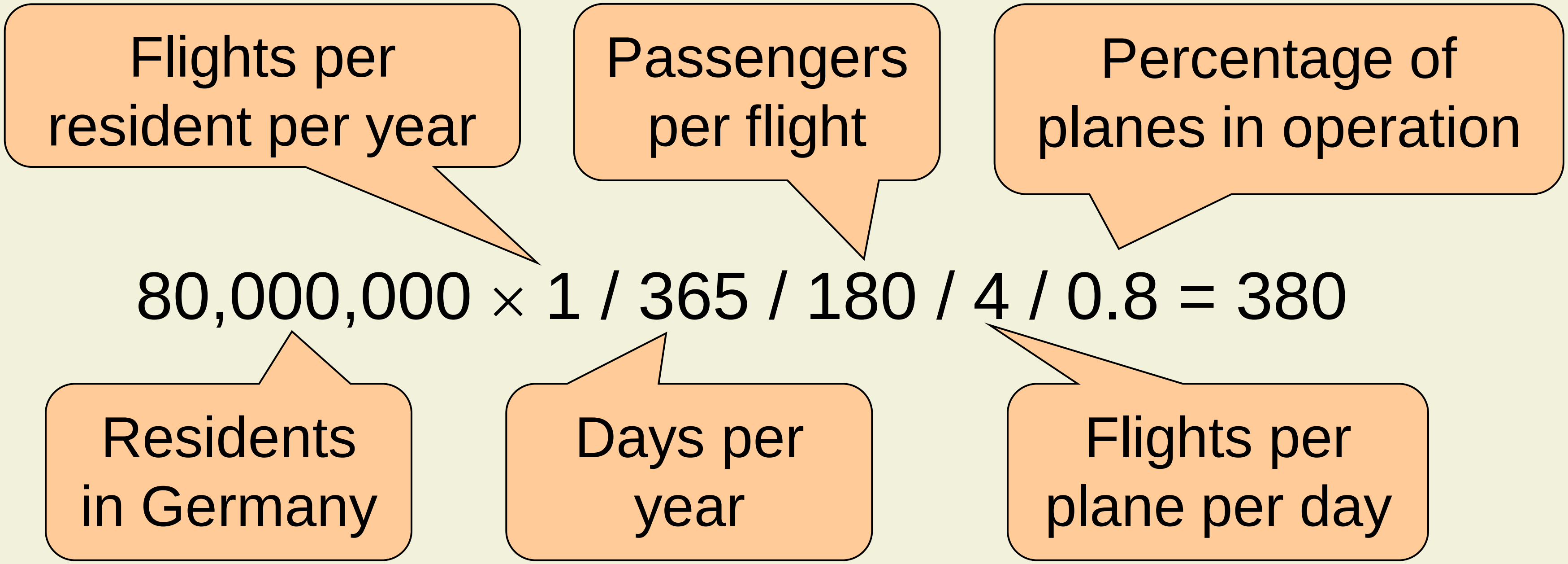
- Rate each element from 0 – 5
    - 0: no influence
    - 1: insignificant influence
    - 2: moderate influence
    - 3: average influence
    - 4: significant influence
    - 5: strong influence
- **Technical complexity factor**
    - TCF = 0.65 + 0.01 × sum
    - Varies between 0.65 and 1.35

Function Point Computation

	Simple	Average	Complex
Inputs	6 x 3 = 18	2 x 4 = 8	3 x 6 = 18
Outputs	7 x 4 = 28	7 x 5 = 35	0 x 7 = 0
Inquiries	0 x 3 = 0	2 x 4 = 8	4 x 6 = 24
Ext. files	9 x 5 = 45	0 x 7 = 0	2 x 10 = 20
Int. files	5 x 7 = 35	2 x 10 = 20	3 x 15 = 45
Unadjusted function points (UFP)			304
Technical complexity factor (TCF)			1.15
Adjusted function points			**350**

- **Adjusted function points**: FP = UFP × TCF

Calibration



Flights per resident per year

Passengers per flight

Percentage of planes in operation

$$80{,}000{,}000 \times 1 / 365 / 180 / 4 / 0.8 = 380$$

Residents in Germany

Days per year

Flights per plane per day

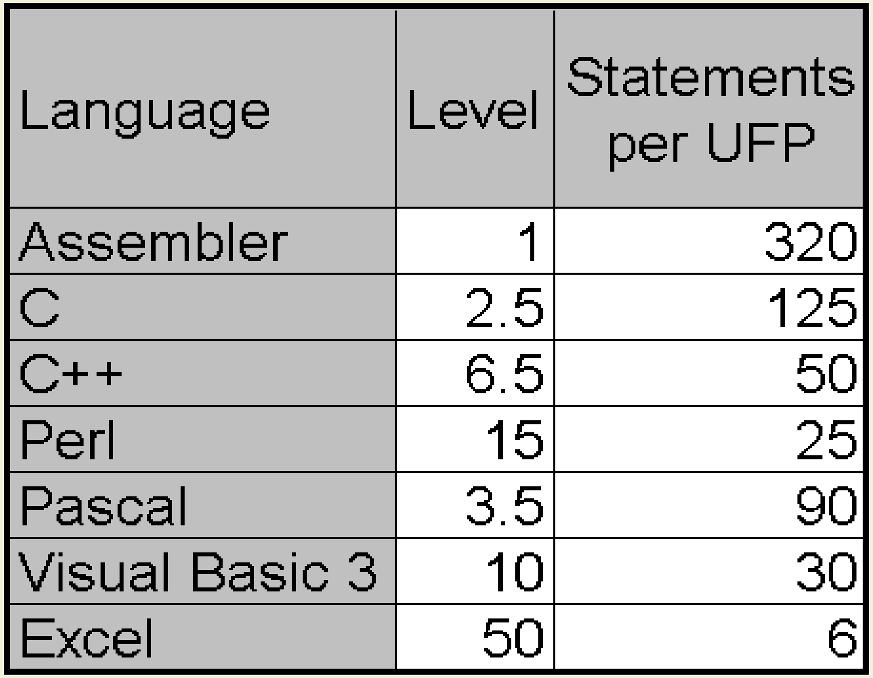
- Assume that model (formula) is correct
- **Calibrate** model based on **comparable** airlines
- Estimate number of residents in the country

Determining Effort and Size

- Empirical value for effort
  - Or use a table

$$\text{Effort} = FP^{1.4} / 150$$

- Empirical value for size
- Huge differences in productivity
  - Factor 10-20 between individual programmers
  - Factor 4 between companies



Language	Level	Statements per UFP
Assembler	1	320
C	2.5	125
C++	6.5	50
Perl	15	25
Pascal	3.5	90
Visual Basic 3	10	30
Excel	50	6

Function Point Analysis: Discussion

Pros

- Based on requirements (instead of code size)
- Can be applied in early project phases
- Can be calibrated (for company, project type)
- Counting standards by "International Function Points User Group"
- Technology-independent

Cons

- Estimation of overall effort (not per phase)
- Tailored towards functional decomposition (rather than OO)
- Tailored towards information systems
- Needs calibration to produce reliable results

Estimation Techniques: Discussion

Empirical Estimation

- Accurate if experts are experienced
- Experts can be strongly biased (over-optimism)

Algorithmic Estimation

- Very accurate if model is calibrated
- Calibration is very difficult and expensive
- Estimation is expensive

- Empirical studies
  - Do not show that uncalibrated algorithmic estimation is, in general, more accurate
  - Show that algorithmic estimation is more accurate than experts who do not have important domain knowledge

Other Estimation Strategies

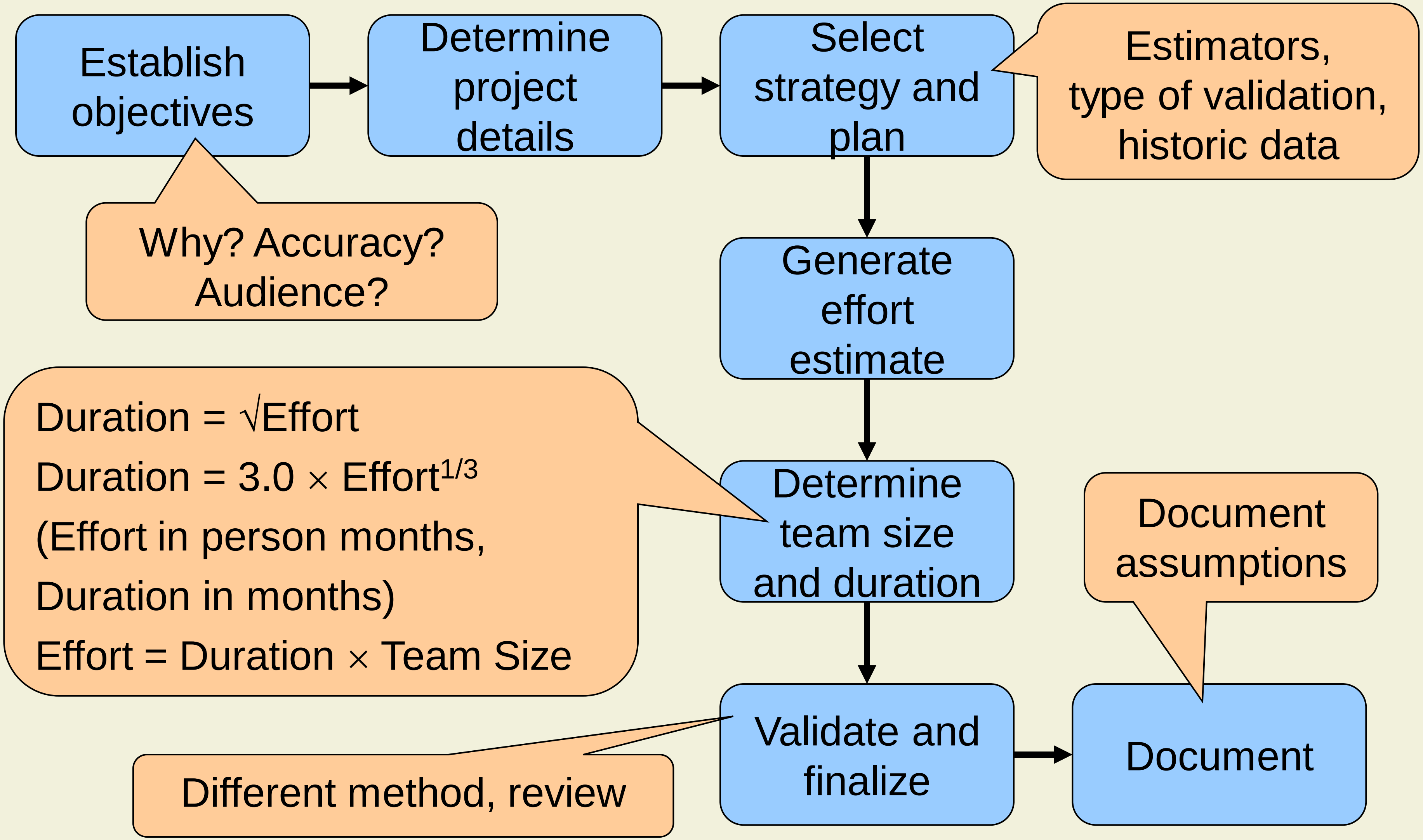
Parkinson's Law

- Work expands to fill the time available
  - Gold plating
- Effort is determined by available resources
- Important for team management

Pricing to win

- Cost is estimated to whatever the customer is willing to spend
- Common strategy to win projects
- Features are negotiated later, constrained by agreed costs
- Costs are fixed, not requirements

Estimating Process


Establish objectives

Determine project details

Select strategy and plan

Estimators, type of validation, historic data

Why? Accuracy? Audience?

Generate effort estimate

Duration = √Effort

Duration = 3.0 × $\text{Effort}^{1/3}$

(Effort in person months, Duration in months)

Effort = Duration × Team Size

Determine team size and duration

Document assumptions

Validate and finalize

Document

Different method, review

12. Project Cost Management

12.1 Software Estimation

12.2 Budgeting

12.3 Earned Value Method

Direct and Indirect Costs

- Direct costs: Costs incurred for the benefit of a specific project
  - Salaries of project staff
  - Equipment bought specifically for the project
  - Travel expenses
- Indirect costs: Costs incurred for the joint benefit over multiple projects (“overhead”)
  - Accounting, quality assurance department
  - Line management
  - Rooms, electricity, heating

Unit Costs

- Projects have to budget for
  - Direct costs
  - A certain share of indirect costs
- Budgets are usually determined by using unit costs
  - Unit cost: Price per unit of a resource
  - Loaded rate: Including indirect costs
  - Unloaded rate: Without indirect costs
- Examples

Loaded day rate for senior IT consultant:	CHF 3.500
Loaded day rate for internal developer:	CHF 1.200

Effort, Duration, and Cost

- Effort: The number of labor units required to complete an activity
- Availability: Time a staff person is able to work
  - For long projects approximately 70% per person
- Productivity: The relative measure of work in a time unit

- Duration = (Effort / Productivity) / (Resources x Availability)
- Cost = (Effort / Productivity) x Unit Cost

Pricing

- The price is often based on the costs and a margin
- Price = Costs / (1 - Margin)
- Example
  - Costs = CHF 1.000.000
  - Margin = 5%
  - Price = CHF 1.052.632
- Price is influenced by
  - Market situation
  - Business strategy

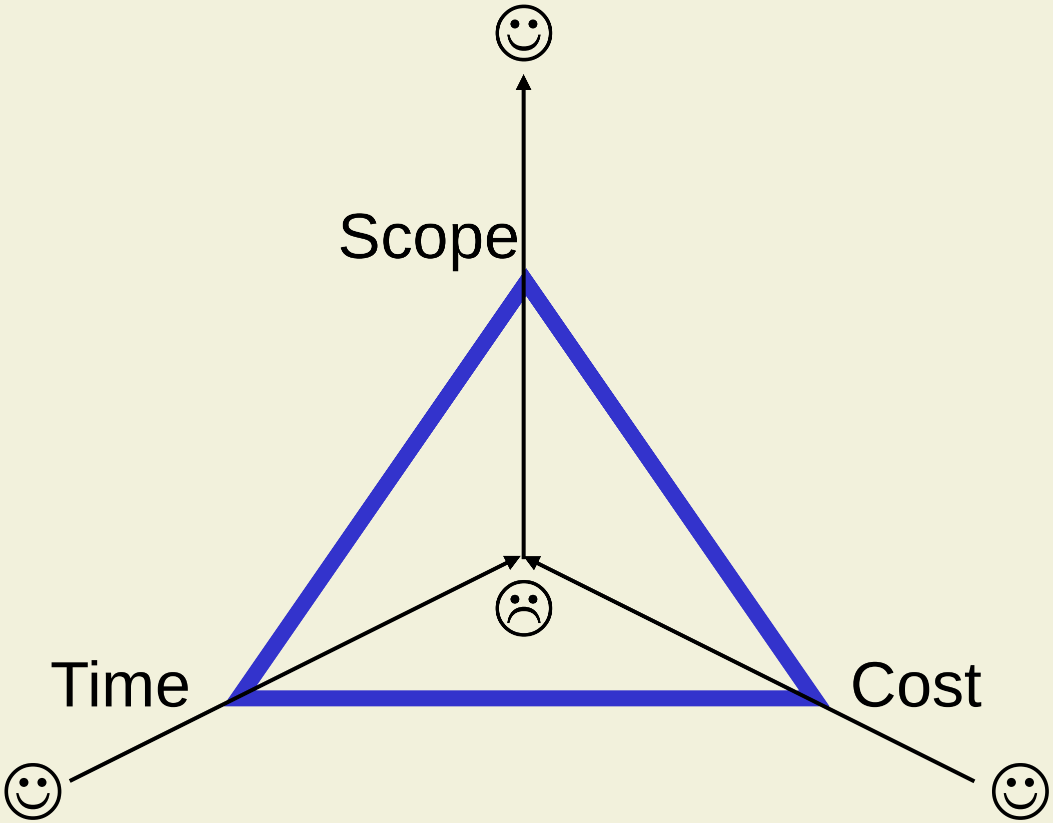
12. Project Cost Management

12.1 Software Estimation

12.2 Budgeting

12.3 Earned Value Method

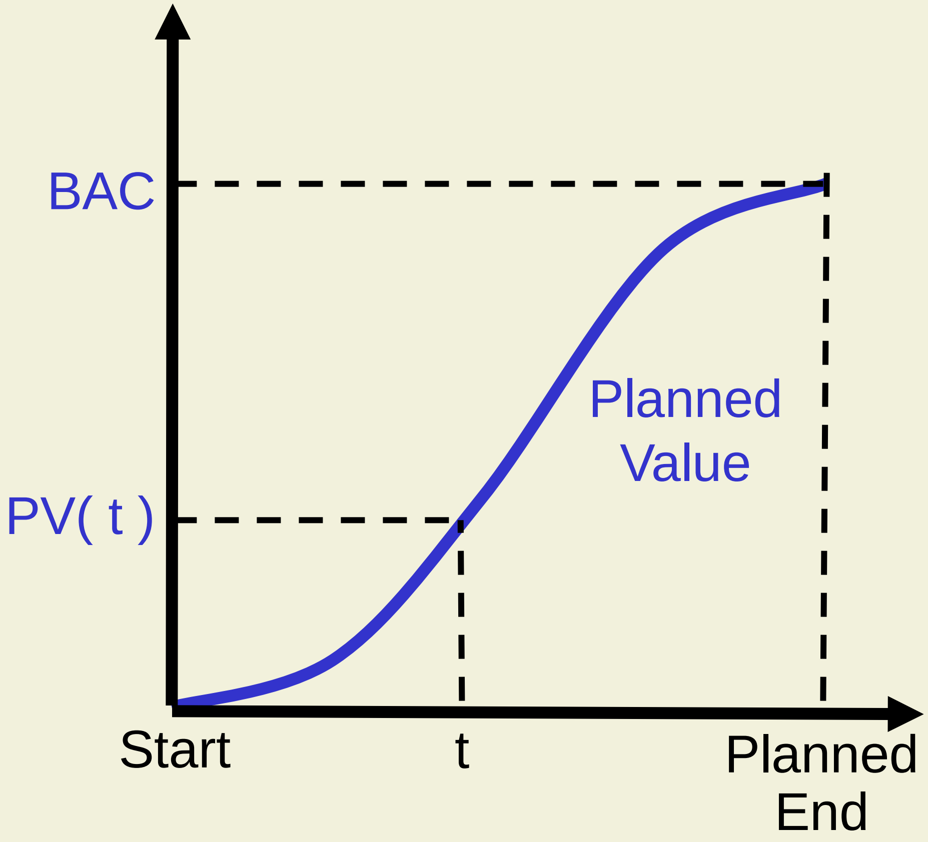
The Triple Constraint



- Project objectives are **equally important**
- Actions in one project area usually affect other areas

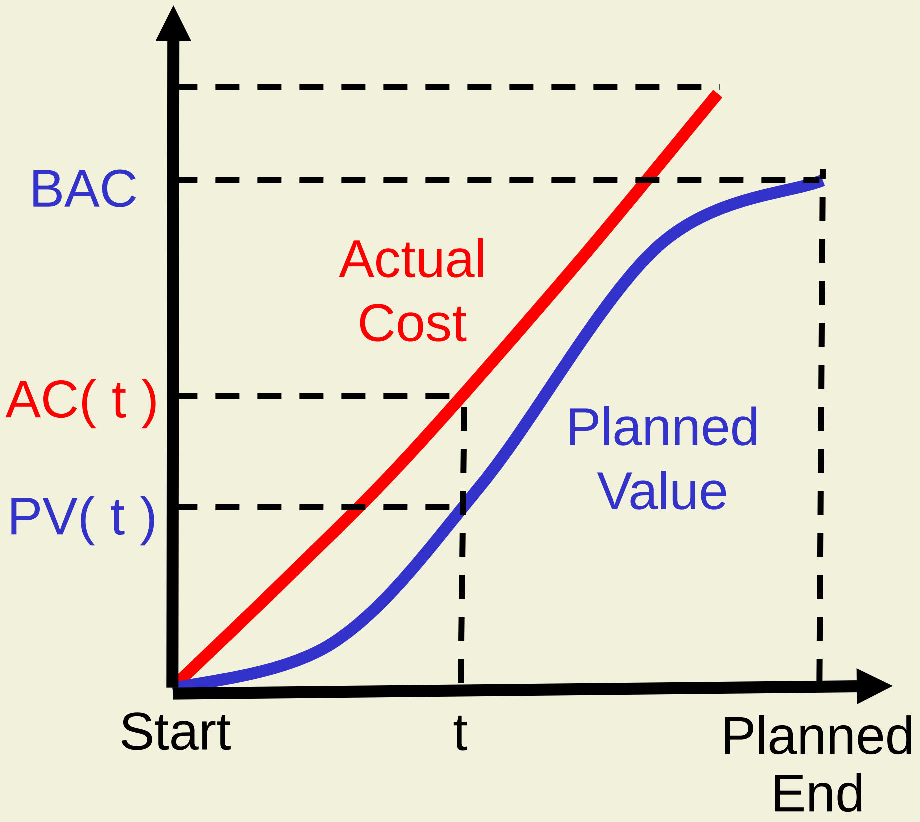
Planned Value (PV)

- The **cumulative** sum of the **approved** cost for activities **scheduled**
- Corresponds to the **cost baseline**
- **Budget at completion** is the estimated baseline total cost: BAC = PV(end)



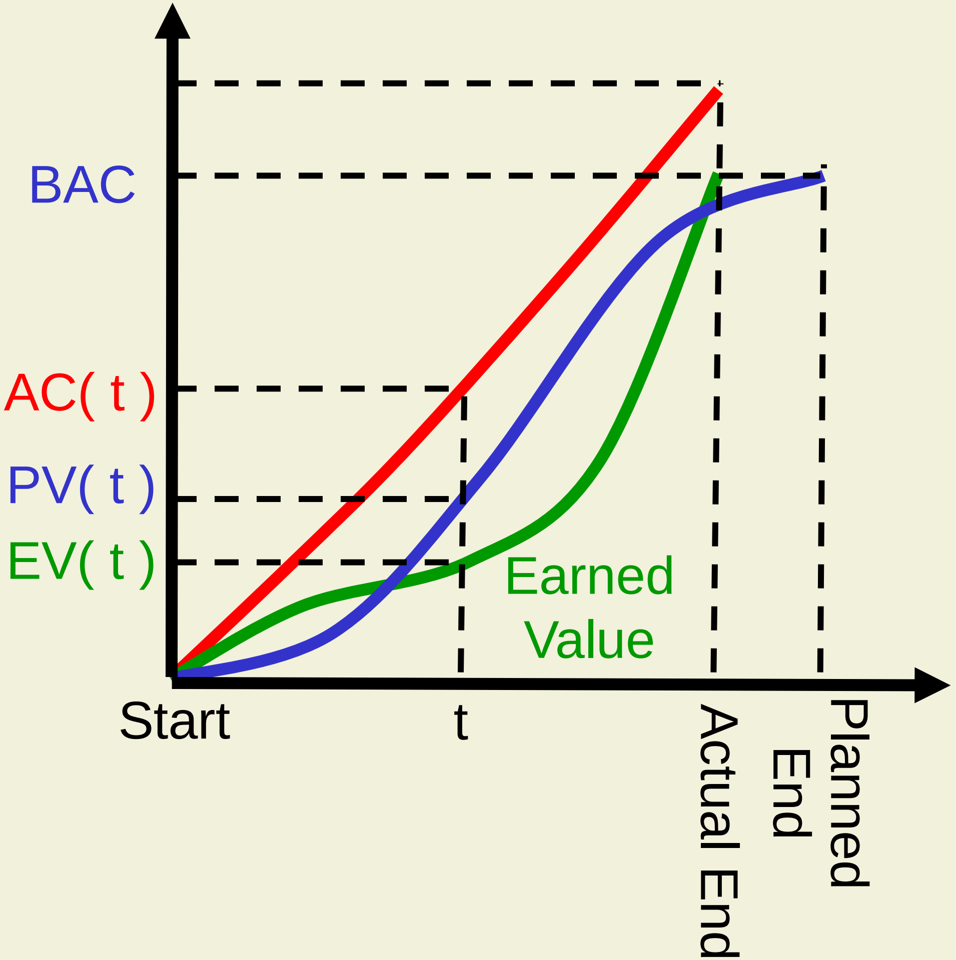
Actual Cost (AC)

- Total **cost incurred** for the project up to a specified date
- The **actual** or **real** cost of work performed
- Contains both direct and indirect cost



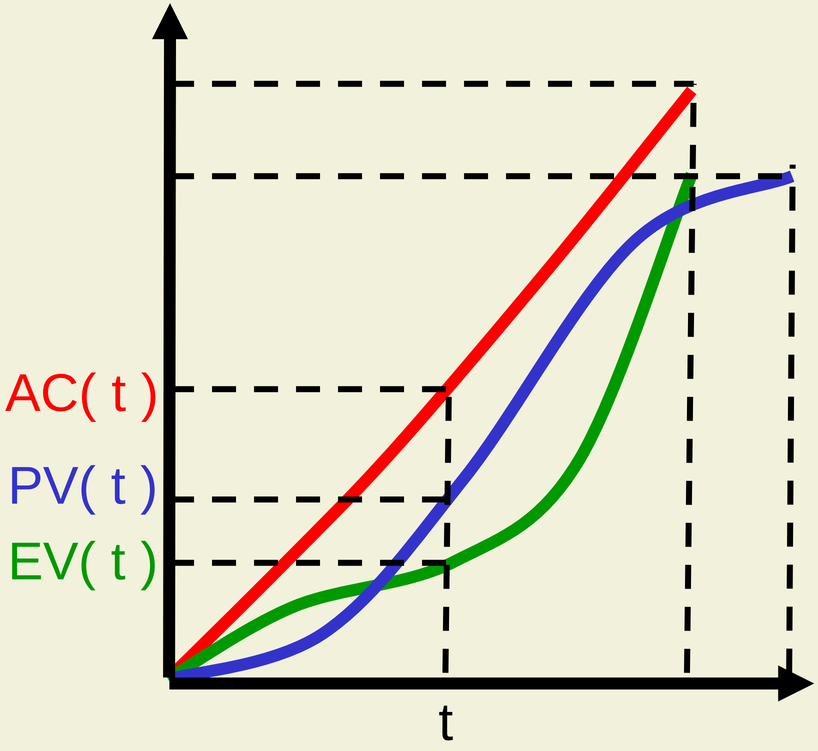
Earned Value (EV)

- The sum of **approved cost estimates** for activities **completed** up to a specified date
- An activity is completed if PV=EV, regardless of the actual cost

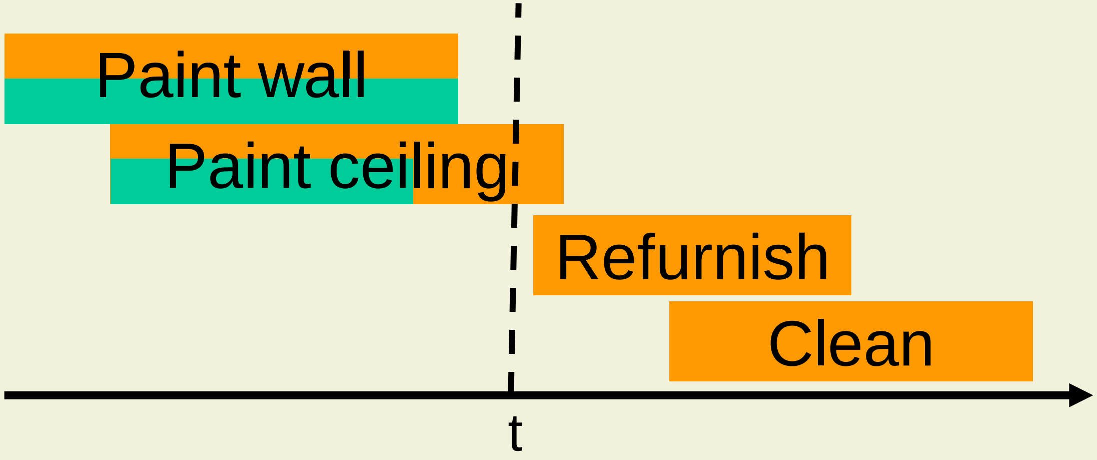


Earned Value Method

- Expresses effort, cost, and time as **monetary value**
    - PV(t): Worth of the activities scheduled (planned)
    - AC(t): Cost spent
    - EV(t): Worth of the activities performed
- Compares the amount of work planned to what was actually accomplished to **determine cost and schedule performance**



Example



Activity	PV(t)	AC(t)	EV(t)
Paint wall	800	1000	800
Paint ceiling	400	300	300
Total	1.200	1.300	1.100

Cost Performance Index (CPI)

- Compares **budgeted cost** of work performed to **actual cost**
- Indicates the **efficiency** of the project

$$\boxed{\mathrm{CPI} = \frac{\mathrm{EV}}{\mathrm{AC}}}$$

- How much do we get out of one Franc we spend?

Activity	PV(t)	AC(t)	EV(t)
Paint wall	800	1000	800
Paint ceiling	400	300	300
Total	1.200	1.300	1.100

$$\mathrm{CPI} = \frac{1.100}{1.300} = 85\%$$

Schedule Performance Index (SPI)

- Compares **work performed** to **work planned**

$$\boxed{SPI = \frac{EV}{PV}}$$

- How fast does the project progress in relation to how fast it is expected to progress?

Activity	PV(t)	AC(t)	EV(t)
Paint wall	800	1000	800
Paint ceiling	400	300	300
Total	1.200	1.300	1.100

$$SPI = \frac{1.100}{1.200} = 92\%$$

Calculated Estimate at Completion

$$CEAC_1 = \frac{BAC}{CPI}$$

- Budget modified by performance
  - If the current variances are **typical for the future**

$$CEAC_2 = AC + BAC - EV$$

- Actual to date plus remaining budget
  - If the current variances are **atypical for the future**

$$CEAC_3 = AC + ETC$$

- Actual plus a new estimate for remaining work
  - If the original estimate was **fundamentally flawed**

Interpreting EV-Indicators

- Typically, indicators are **stable after 20%** of the project duration

- CPI > 1: Project is in budget
- CPI < 1: Project is over budget
- SPI > 1: Project is ahead of schedule
- SPI < 1: Project is behind schedule

Golden Rules of Earned Value

- Rule 1: Earned value should be verified by **physically examining** the **work product** associated with the activity
- Rule 2: For unfinished activities, earned value estimates are usually just a guess. Apply one of the following rules consistently
    - **50/50 Rule**: A task is considered 50% complete when it begins and 100% only when it is completed
    - **20/80 Rule**: A task is considered 20% complete when it begins and 100% only when it is completed
    - **0/100 Rule**: A task does not get credit for partial completion, only for full completion