Politecnico
di Milano

Project management lecture 2

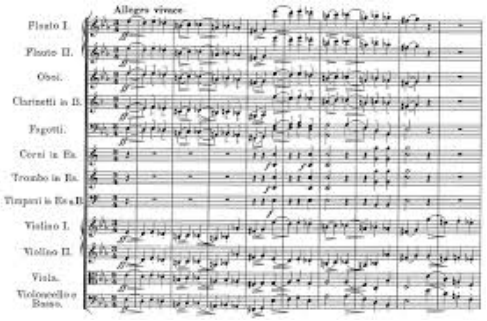
Enrico Gargantini

Project Management

The five processes of project management

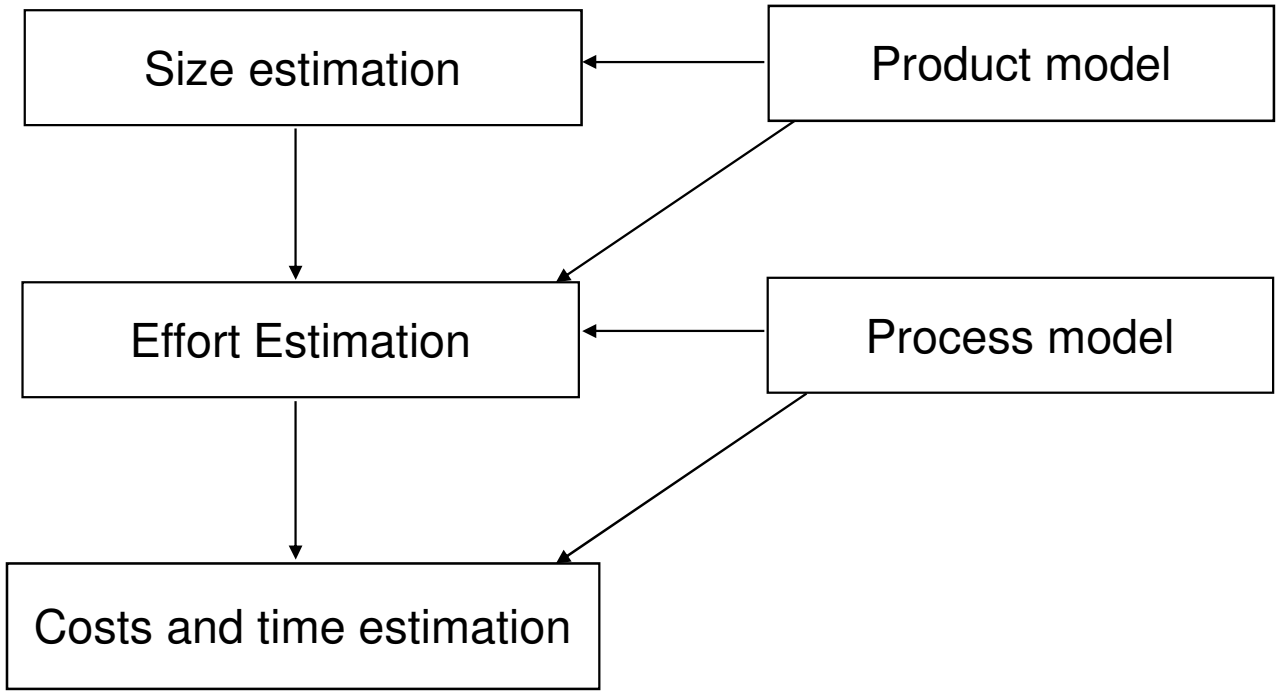
1. Initiating
2. Planning
    1. Cost and effort estimation
3. Executing
4. Monitoring and Controlling
5. Closing

SW Project Management like an orchestra

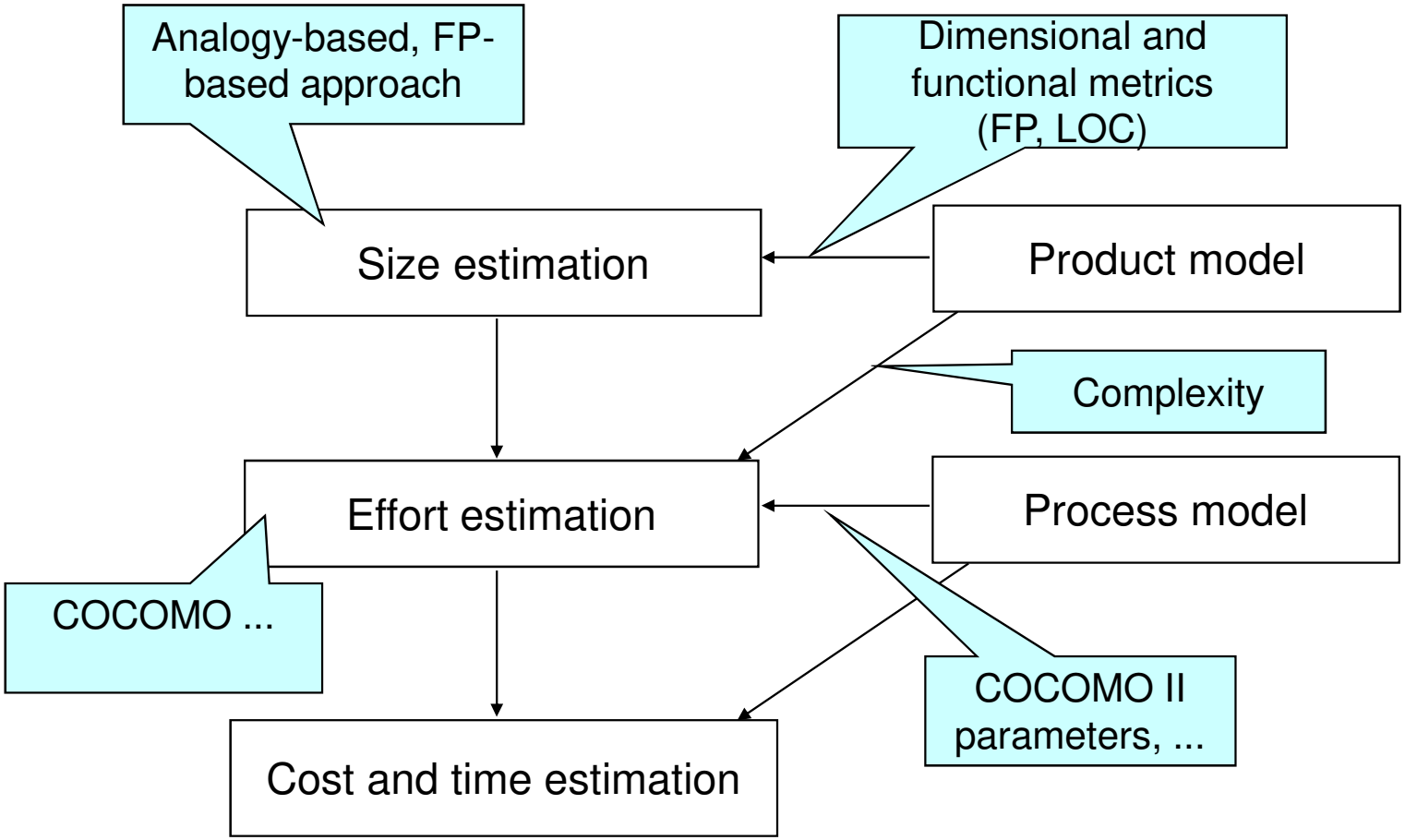


SW Project Management like an orchestra

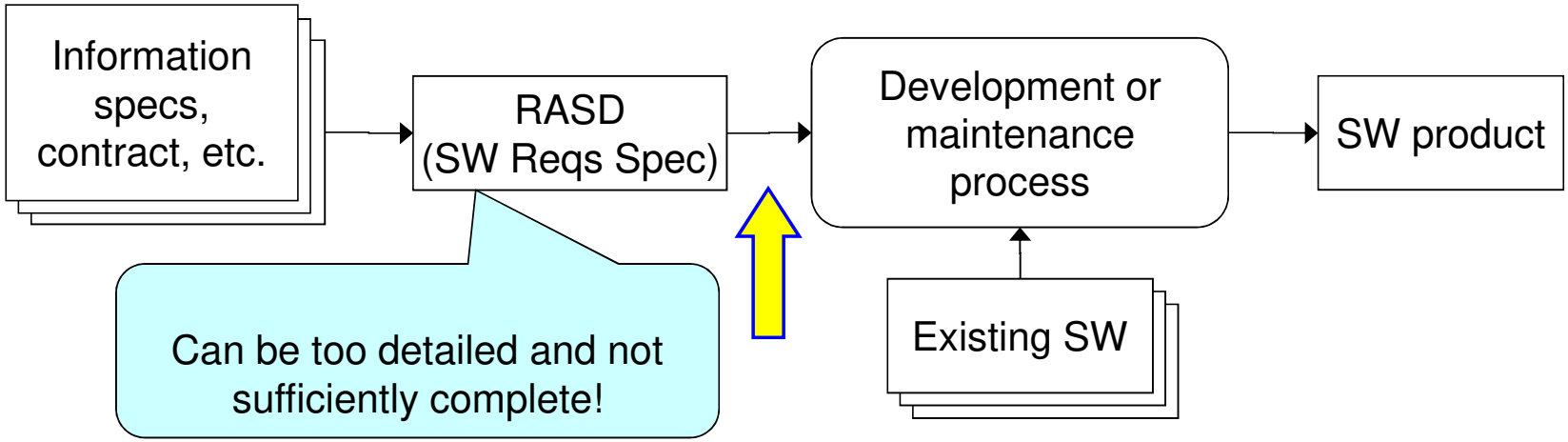
A possible estimation procedure



Techniques to support the estimation procedure



RASD-based size estimation



- Options:
  - Calculate the number of **Function Point**
    - This can be used directly as a measure of the software dimension
  - Estimate the LOC
    - By converting the FP into LOCs
    - Directly
  - Any estimation has to be completed by a complexity analysis

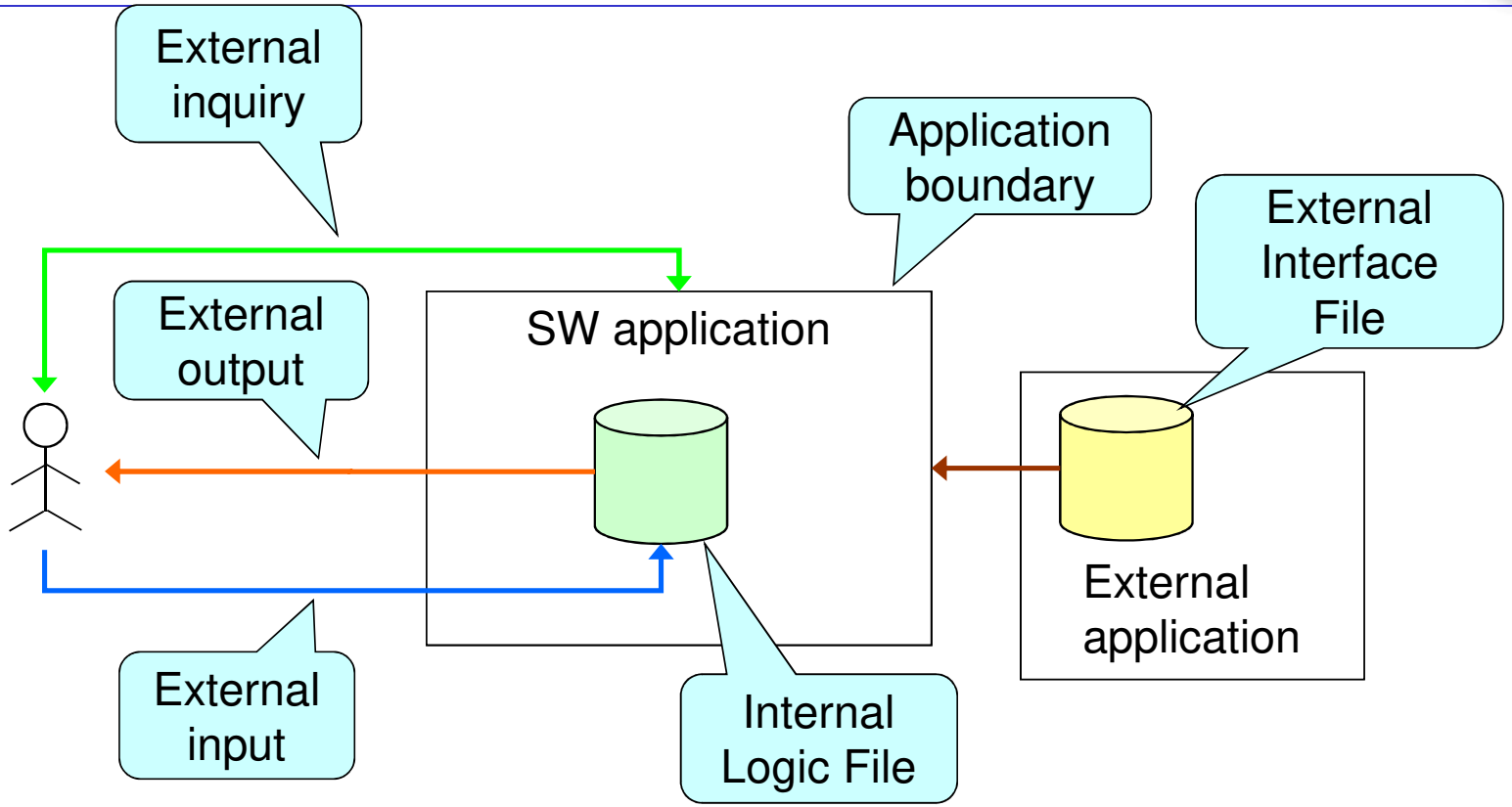
Function points

- The Function Points approach has been defined in 1975 by Allan Albrecht (IBM)
- Basic assumption
  - the dimension of software can be characterized based on the functionalities that it has to offer

What are Function Points (FP)?

- Based on a combination of program characteristics
  - Data structures;
  - Inputs and outputs;
  - Inquiries;
  - External interfaces;
- A weight is associated with each of these FP counts; the total is computed by multiplying each “raw” count by the weight and summing all partial values:
- UFP = $\sum$ (number of elements of a given type * weight)

Function Types (1)



Function Types (2)

- The Albrecht's method identifies and count the number of function types
- They, all together, constitute the "external representation" of an application, that is, its functionality
- Function types
  - Internal Logical File (ILF): homogeneous set of data used and managed by the application
  - External Interface File (EIF): homogeneous set of data used by the application but generated and maintained by other applications

Function Types (3)

- External Input
  - Elementary operation to elaborate data coming from the external environment
- External Output
  - Elementary operation that generates data for the external environment
    - It usually includes the elaboration of data from logic files
- External Inquiry
  - Elementary operation that involves input and output
    - Without significant elaboration of data from logic files

How was the approach defined?

- Albrecht has considered 24 applications
  - 18 COBOL, 4 PL/1, 2 DMS
  - …ranging from 3000 to 318000 LOC,
  - …from 500 to 105200 person hours
- Based on these he has defined the number of FP as the sum of Function Types weighted in order to correlate them to the development effort

Weighting function points

Complexity is evaluated based on the characteristics of the application

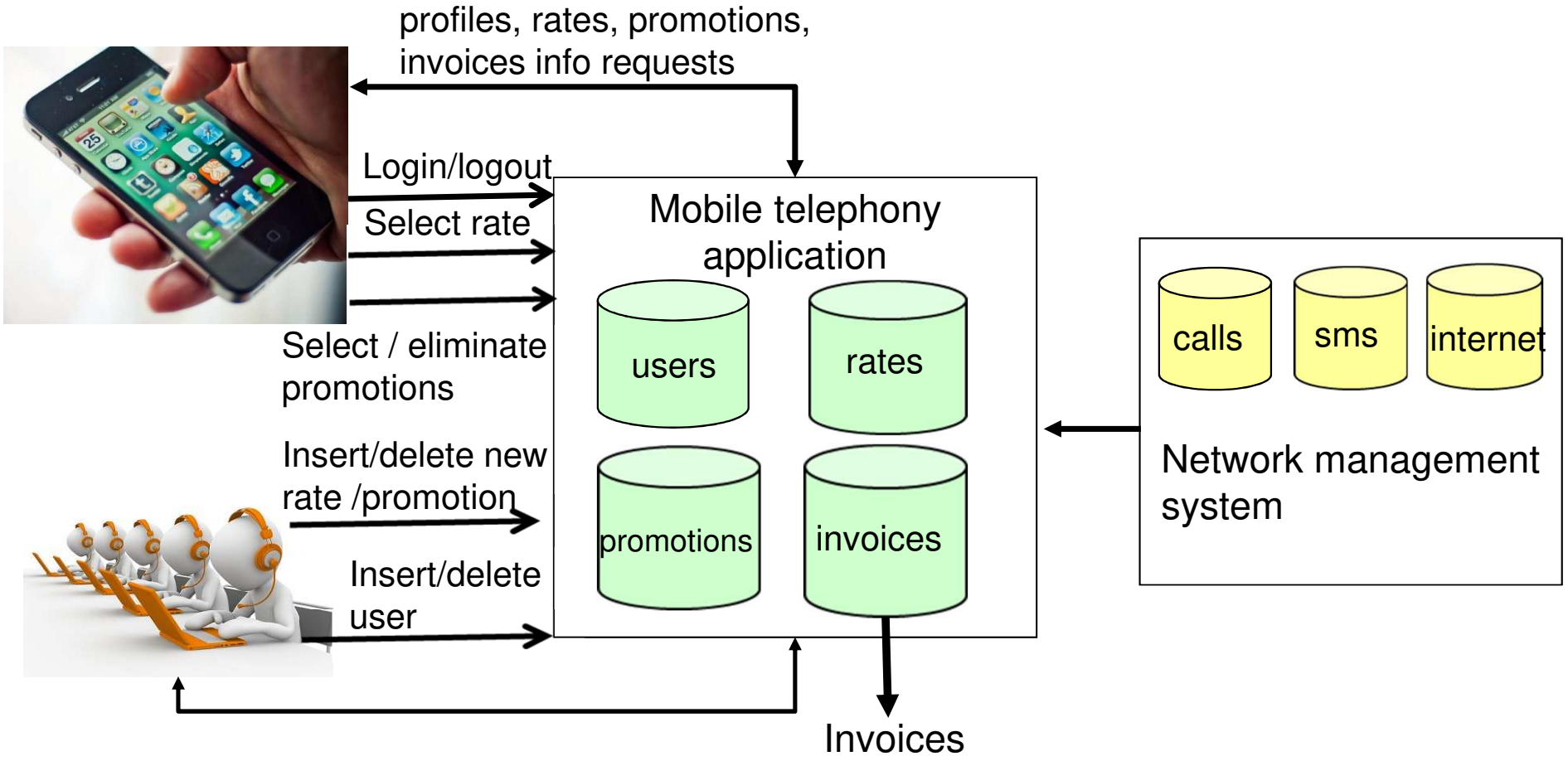
Function Types	Weight		
	Simple	Medium	Complex
N. Inputs	3	4	6
N. Outputs	4	5	7
N. Inquiry	3	4	6
N. ILF	7	10	15
N. EIF	5	7	10

- We obtain the Unadjusted Function Points (UFP)

Example

- Calculate FPs for an application managing a mobile telephony service. The application:
  - Manages data about both users and the different types of rates. The system will store the user data (phone number, personal data, the type of rate, any applying promotion). Each rate type is characterized by: call cost, cost of text messages, cost of surfing the Internet. Promotions change some of the cost items only for a well-defined period of time.
  - Allows users to access their information, change rates, and activate promotions.
  - Manages user billing based on information retrieved from the network management system. This last system sends to the service management system data concerning the voice calls (call duration, and the user location in case of roaming), SMSs (the information that messages have been sent, and the corresponding user location in case of roaming), and Internet surfing (connection duration, amount of downloaded data, user location) of each user.

Example


profiles, rates, promotions,
invoices info requests
Login/logout
Select rate
Select / eliminate
promotions
Insert/delete new
rate /promotion
Insert/delete
user
Mobile telephony
application
users
rates
promotions
invoices
calls
sms
internet
Network management
system
Invoices

ILFs

- The application stores the information about
  - *users*,
  - *rates*, and
  - *promotions*.
  - As the application also manages billing information, it will have to produce *invoices*.
- Each of these entities has a simple structure as it is composed of a small number of fields. Thus, we can decide to adopt for all the four the simple weight.
- Thus, 4 x 7 = 28 FPs concerning ILFs.

ELFs

- The application manages the interaction with the network management system for acquiring information about
  - *calls*,
  - *SMSs*, and
  - *Internet usage*.
- Three entities with a simple structure. Thus, we decide to adopt a simple weight for the three of them.
- We get 3 x 5 = 15 FPs.

External inputs (1)

- The application interacts with the customer:
  - Login/logout: these are simple operations, so we can adopt the simple weight for them. 2 x 3 = 6 FPs
  - Select a rate: this operation involves two entities, the rate and the user. It can still be considered simple, so, again we adopt the simple weight. 1 x 3 = 3 FPs
  - Select/eliminate a promotion: These two operations involve three entities, the user, the rate and the promotion. As such, we can considered them of medium complexity. 2 x 4 = 8 FPs

External inputs (2)

- The application also interacts with the company operators to allow them to:
  - Insert/delete information about a new rate or promotion.
  - Insert/delete information about a new user.
- Both the above operations can be consider of average complexity. 4 x 4 = 16 FPs
- In summary, we have FPs = 6+3+8+16 = 33

External inquiries

- The application allows customers to request information about
  - their profiles
  - the list of rates, promotions, and invoices that have been defined for them.
- The application also allows the operator to visualize the information of all customers.
- In summary, we have 5 different external inquiries that we can consider of medium complexity. Thus, 5 x 4 = 20 FPs.

External outputs

- The application allows the creation of invoices.
- This is a quite complex operation as it needs to collect information from IFLs and EFLs.
- Thus, we can apply the weight for the complex cases: FP = 1 x 7.

Total number of FPs

- ILFs: 28
- ELFs: 15
- External Inputs: 33
- External Inquiries: 20
- External Outputs: 7
- Total: 103

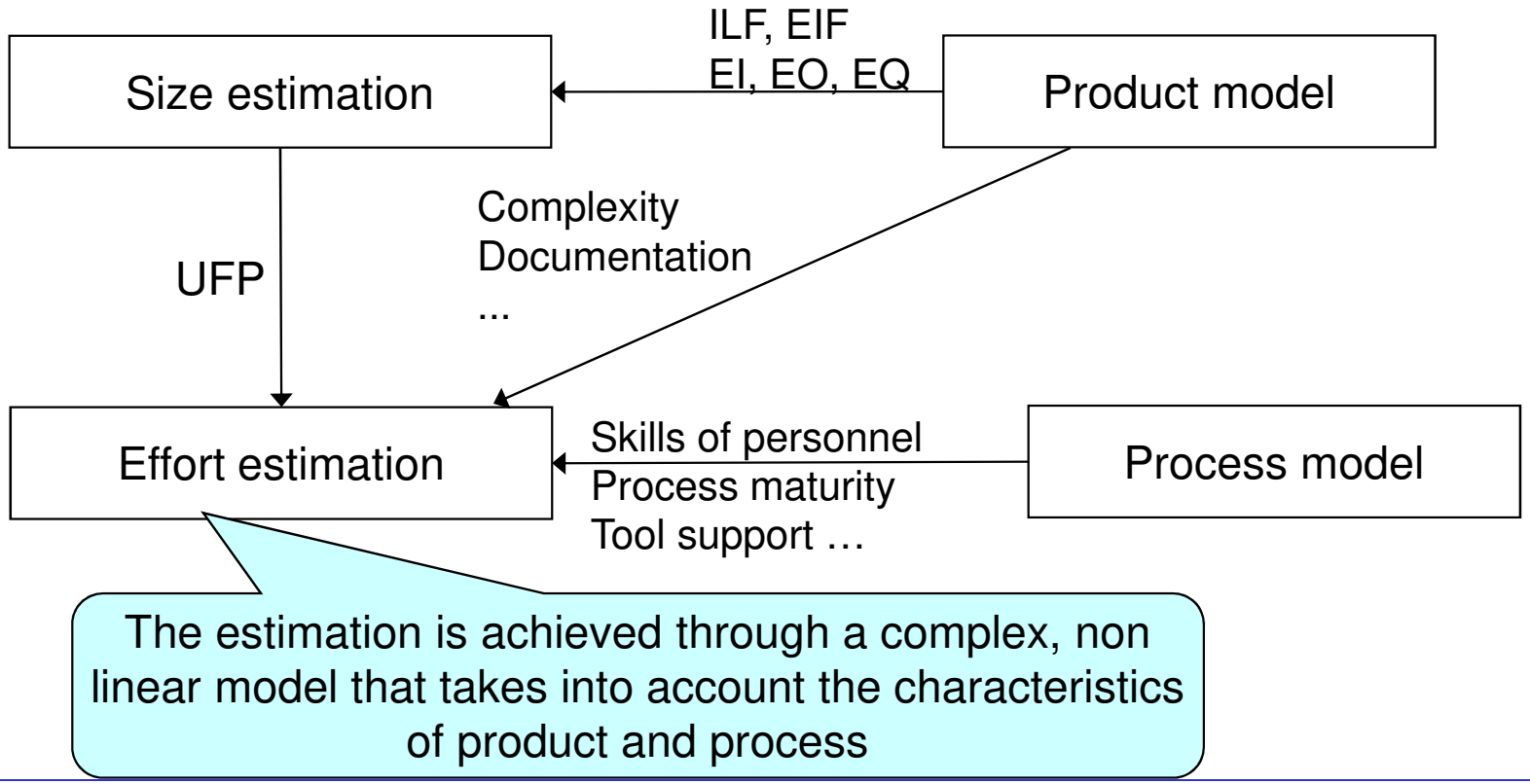
Final remarks

- The function point count is modified by complexity of the project
- FPs can be used to estimate LOC depending on the average number of LOC per FP for a given language
  - LOC=AVC*number of function points;
    AVC is a language-dependent factor varying from 200-300 for assembly language to 2-40 for a 4GL;
  - See here for a possible mapping http://www.qsm.com/resources/function-point-languages-table
- FPs are very subjective. They depend on the estimator

The International Function Point Users Group

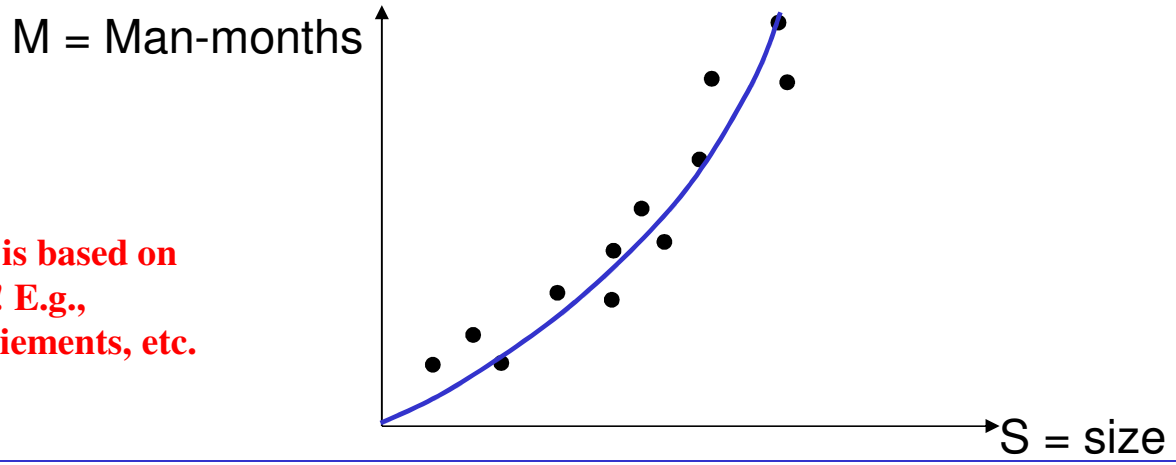
- Mission: takes care of the evolution of FPs
- Produces a manual for supporting the adoption of the FP approach
- Defines various versions of FP to apply them outside the initial context
- Offers examples useful as a reference
- http://www.ifpug.org/

COCOMO 81 (COnstructive COst Model v.1981): positioning



COCOMO 81: How was it defined?

- Barry Boehm has adopted a statistical approach to calculate a basic formula*;
- Given the size and effort of a number of homogeneous projects (same complexity) Boehm has been looking for the best correlation;
  - The linear regression applied to the logarithms of variables has offered the best results:



***COCOMO 81 model is based on outdated assumptions! E.g., waterfall, stable requriements, etc.**

COCOMO II

- Evolution of COCOMO 81 that takes into account some of the new characteristics of software development activities
- Focused on effort estimation in two cases
  - *Post-Architecture* when we are extending an existing product/product line
    - We have detailed information on cost driver inputs, and enables more accurate cost estimates.
  - *Early Design* when we do not have clear information on the architecture of the system
    - For instance, we are exploring different architectural alternatives or incremental development strategies.
    - We do not have detailed information. Thus, the estimation will be less accurate.

The main COCOMO formula: Effort Equation

- $\mathbf{PM = A \times Size^{E} \times \prod_{1<=i<=n} EM_i}$
  - Where
    - A = 2.94. This approximates a productivity constant in PM/KSLOC (Person-Months/Kilo-Source Lines of Code)
    - Size is the estimated size of the project in KSLOC. It can be deducted from UFP
    - EM is for Effort Multiplier. The method offers an approach to derive them from *Cost Drivers*
    - E is an aggregation of five *Scale Factors*

[Coming up next… computing Scale and Cost Drivers!]

Scale factors

- *Precedentedness*: High if a product is similar to several previously developed projects
- *Development Flexibility*: High if there are no specific constraints to conform to pre-established requirements and external interface specs
- *Architecture / Risk Resolution*: High if we have a good risk management plan, clear definition of budget and schedule, focus on architectural definition
- *Team Cohesion*: High if all stakeholders are able to work in a team and share the same vision and commitment.
- *Process Maturity*: Refers to a well known method for assessing the maturity of a software organization, CMM, now evolved into CMMI (see additional material)

Scale factors

Scale Factors	Very Low	Low	Nominal	High	Very High	Extra High
PREC	thoroughly unprecedented	largely unprecedented	somewhat unprecedented	generally familiar	largely familiar	thoroughly familiar
SF_j:	6.20	4.96	3.72	2.48	1.24	0.00
FLEX	rigorous	occasional relaxation	some relaxation	general conformity	some conformity	general goals
SF_j:	5.07	4.05	3.04	2.03	1.01	0.00
RESL	little (20%)	some (40%)	often (60%)	generally (75%)	mostly (90%)	full (100%)
SF_j:	7.07	5.65	4.24	2.83	1.41	0.00
TEAM	very difficult interactions	some difficult interactions	basically cooperative interactions	largely cooperative	highly cooperative	seamless interactions
SF_j:	5.48	4.38	3.29	2.19	1.10	0.00
PMAT	The estimated Equivalent Process Maturity Level (EPML) or					
	SW-CMM Level 1 Lower	SW-CMM Level 1 Upper	SW-CMM Level 2	SW-CMM Level 3	SW-CMM Level 4	SW-CMM Level 5
SF_j:	7.80	6.24	4.68	3.12	1.56	0.00

- $E = B + 0.01 \times \sum_{1<=j<=5} SF_j$, where $B = 0.91$

Scale factors

Scale Factors	Very Low	Low	Nominal	High	Very High	Extra High
PREC	thoroughly unprecedented	largely unprecedented	somewhat unprecedented	generally familiar	largely familiar	thoroughly familiar
SF_j:	6.20	4.96	3.72	2.48	1.24	0.00
FLEX	rigorous	occasional relaxation	some relaxation	general conformity	some conformity	general goals
SF_j:	5.07	4.05	3.04	2.03	1.01	0.00
RESL	little (20%)	some (40%)	often (60%)	generally (75%)	mostly (90%)	full (100%)
SF_j:	7.07	5.65	4.24	2.83	1.41	0.00
TEAM	very difficult interactions	some difficult interactions	basically cooperative interactions	largely cooperative	highly cooperative	seamless interactions
SF_j:	5.48	4.38	3.29	2.19	1.10	0.00
PMAT	The estimated Equivalent Process Maturity Level (EPML) or					
	SW-CMM Level 1 Lower	SW-CMM Level 1 Upper	SW-CMM Level 2	SW-CMM Level 3	SW-CMM Level 4	SW-CMM Level 5
SF_j:	7.80	6.24	4.68	3.12	1.56	0.00

- $E = B + 0.01 \times \sum_{1<=j<=5} SF_j$, where $B = 0.91$

Empirically-proven quantities!

Cost Drivers for post-architecture

- The selection of used cost drivers depend on whether we are in the case of post-architecture or early design.
- For post-architecture 16 different factors grouped in four categories
- **Product factors**
  - Required Software Reliability (RELY)
  - Data Base Size (DATA)
  - Product Complexity (CPLX)
  - Developed for Reusability (RUSE)
  - Documentation Match to Life-Cycle Needs (DOCU)
- **Platform factors**
  - Execution Time Constraint (TIME)
  - Main Storage Constraint (STOR)
  - Platform Volatility (PVOL)

Cost Drivers for post-architecture

- **Personnel Factors**
  - Analyst Capability (ACAP)
  - Programmer Capability (PCAP)
  - Personnel Continuity (PCON)
  - Applications Experience (APEX)
  - Platform Experience (PLEX)
  - Language and Tool Experience (LTEX)
- **Project Factors**
  - Use of Software Tools (TOOL)
  - Multisite Development (SITE)
- **General Factor**
  - Required Development Schedule (SCED)

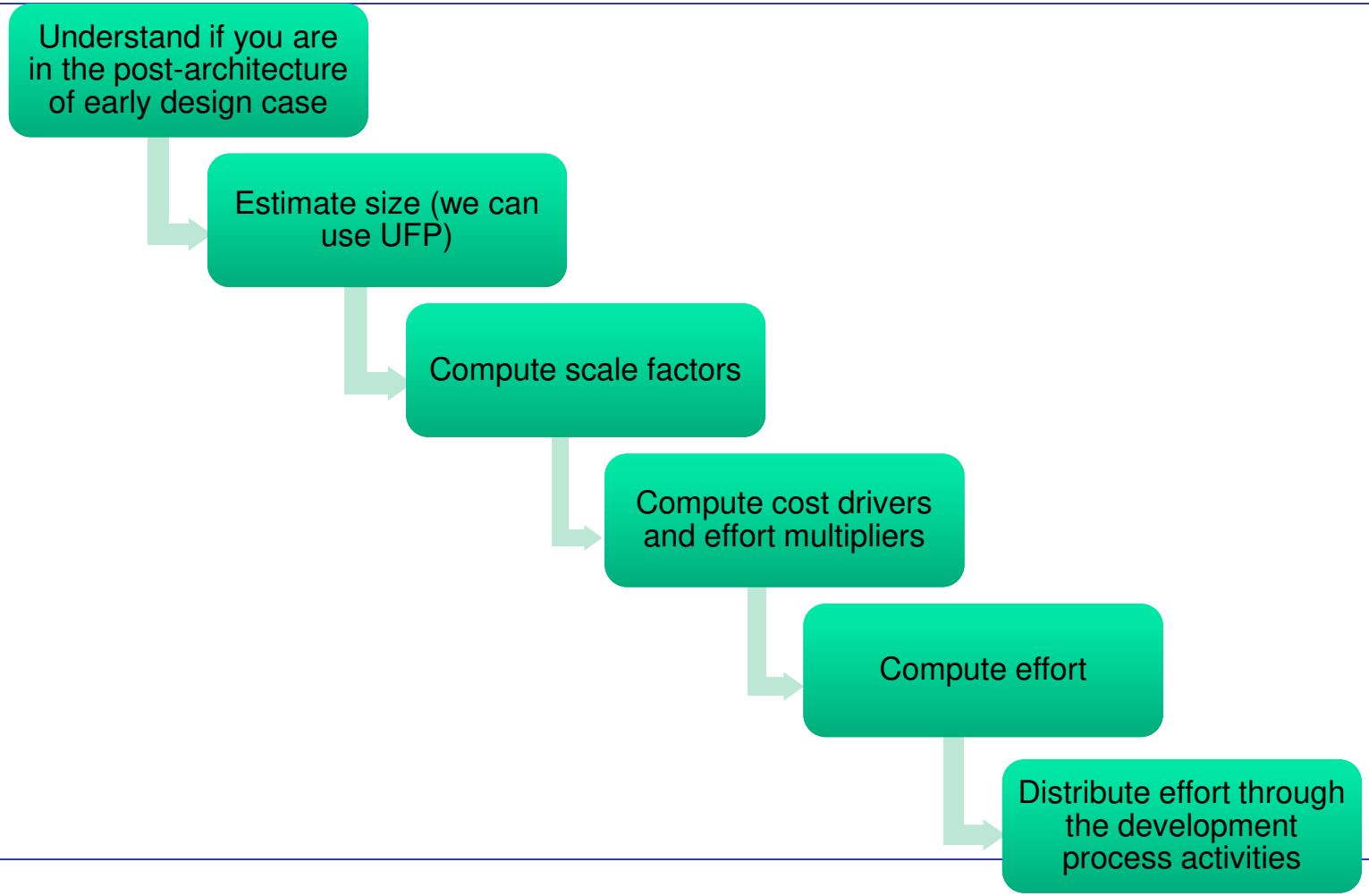
Cost Drivers for early-design

Early Design Cost Driver	Counterpart Combined Post-Architecture Cost Drivers
PERS	ACAP, PCAP, PCON
RCPX	RELY, DATA, CPLX, DOCU
RUSE	RUSE
PDIF	TIME, STOR, PVOL
PREX	APEX, PLEX, LTEX
FCIL	TOOL, SITE
SCED	SCED

From Cost Drivers to Effort Multipliers: an example

RELY Descriptors:	slight inconven-ience	low, easily recoverable losses	moderate, easily recoverable losses	high financial loss	risk to human life	
Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	0.82	0.92	1.00	1.10	1.26	n/a

The COCOMO application process



What do we get from COCOMO?

- Never forget the statistical nature of COCOMO.
- When we apply it and get an estimation of effort, the resulting number is based on the knowledge of the current practice that is encapsulated in the model
- In other words, if the project is similar to the ones that have been considered for the definition of the model, then the computed effort is likely to be correct
  - Sometimes we need to calibrate the model

Executing process

Launch the project : kick off meeting

Acquire and manage project team (internal and external resources)

Acquire the required equipment and materials and external services

Execute the plans (communication, change, quality)

Perform the work identified in the WBS

Perform controlling and monitoring activities

Executing process : daily instruments

- the plan review towards the baseline
- the daily issue register
- the daily log
- the risk register
- the quality register
- The status reports (according with the communication procedure – time driven)
- Event notifications (according with the communication procedure – event driven)

Monitoring and controlling process

Monitoring consists of collecting data about where the project stands, since projects never stick to the initial plans due to changes, problems, etc.

Controlling is where you implement corrections to get your project back on track.

Monitoring process – data gathering

Gather up to date data (actual dates, time worked, money spent)

In detail :

- Track when tasks start
- Track actual work hours or actual duration
- Track how much work or duration remains
- Explore additional costs

Update the schedule (the original is called baseline)

Monitoring schedule and cost

Compare your actual schedule with the baseline schedule : early warning signs

- Incomplete tasks running late
- Should have started but haven't
- Haven't completed as much work as planned

Compare actual costs with assigned budget

Monitoring process - scope

Managing scope is key in project success.

Scope can creep can be due to

- Unclear scope > pay attention to scope definition
- Requests for change > change management plan
- Changed business needs > renegotiate scope

Monitoring process - risks

Monitor risks and in the case they become reality step into the risk response defined in the risk management plan (avoid, mitigate, transfer, plan contingency, accept, enhance)

Monitoring process – Earned Value Analysis

The financial worth a project has earned based on work completed > it reduces all values to financial values so they can be compared.

Monitoring process – Earned Value Analysis

It is based on :

Budget at completion (BAC): total budget for the project
Planned value (PV): budgeted cost of work planned
Earned value (EV): budgeted cost of work performed
Actual cost (AC): actual cost for the completed work

Costs and work should be completed as of today.

Earned Value Analysis : os servers upgrade

Scope : upgrade 4 servers

Time : 4 weeks

Budget : 400 EUR

Schedule

time	scope	budget (EUR)
week 1	server 1	100
week 2	server 2	100
week 3	server 3	100
week 4	server 4	100

Earned Value Analysis : os servers upgrade

Monitoring at the end of week 2

Actual spent : 150

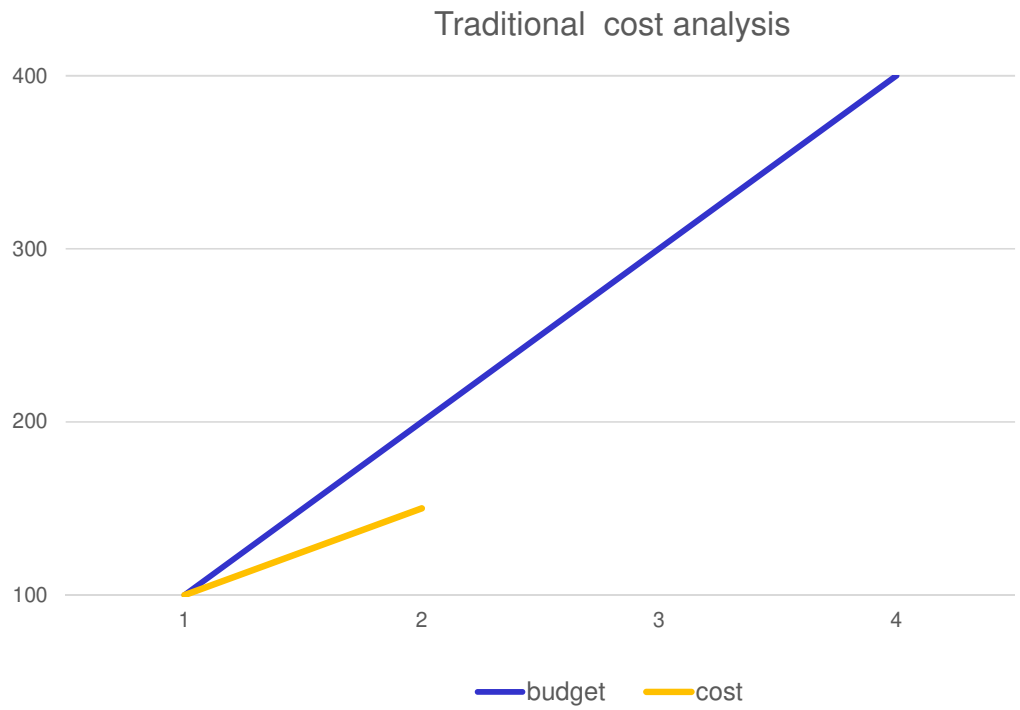
1 server upgraded

Traditional PM :

Actual spent 150 ; Planned spent 200 > under budget

No information on the schedule

Earned Value Analysis : os servers upgrade



Under budget (false)
No information on schedule

Earned Value Analysis : os servers upgrade

Monitoring at the end of week 2

Actual spent : 150

1 server upgraded

Earned value (Budgeted cost of work performed):

1 server upgrade, budgeted cost 100 euro

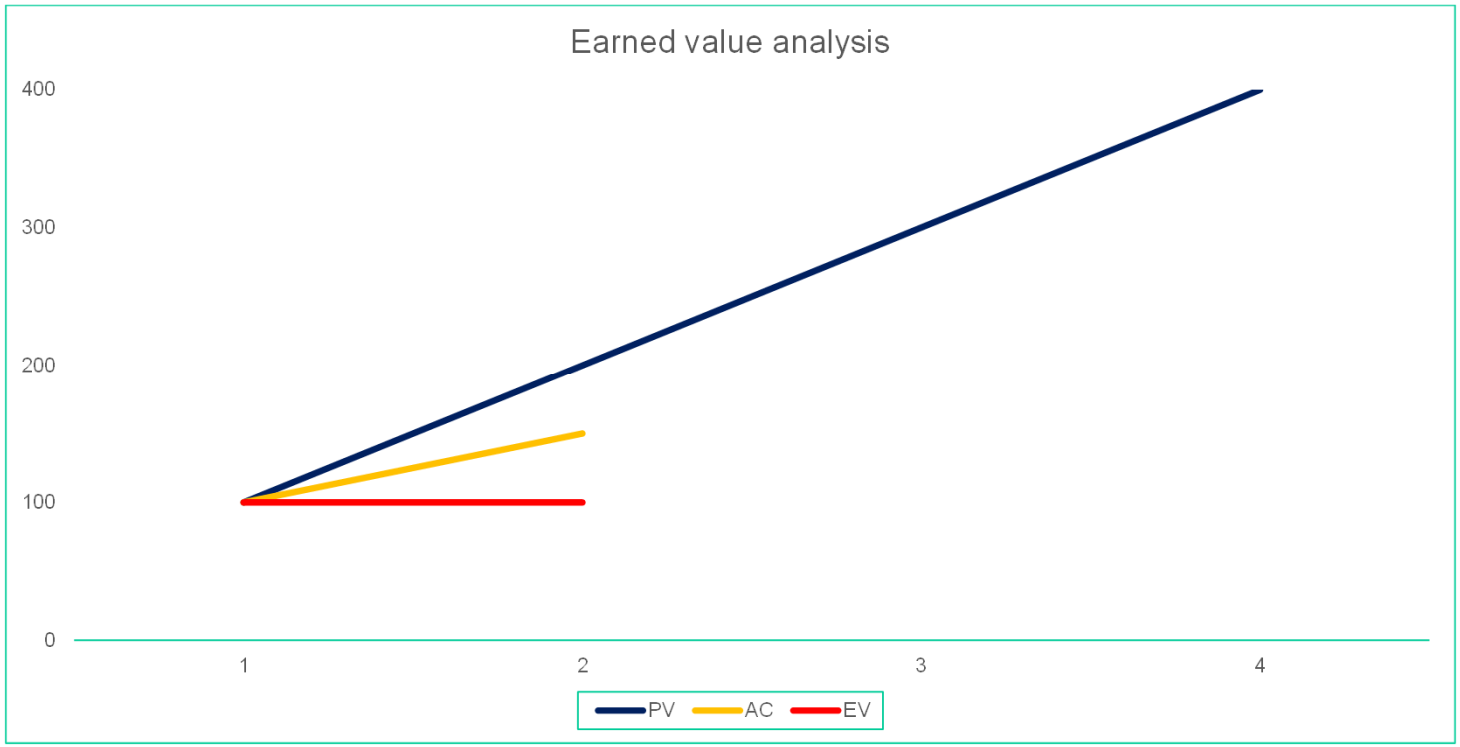
EV = 100

Planned value (Budgeted cost of work planned)

2 servers , 100 euro each > PV = 200

Actual spent : 150 euro

Earned Value Analysis : os servers upgrade



Over budget (EV < AC)
Behind schedule (EV < PV)

Monitoring process – Earned Value Analysis

Schedule point of view

Schedule variance : SV = EV – PV

Schedule performance index : SPI = EV / PV

Cost point of view

Cost variance : CV = EV – AC

Cost performance index : CPI = EV/AC

Monitoring process – Earned Value Analysis

Estimate at completion

Assumption 1 : continue to spend at the same rate

EAC = BAC/CPI

Assumption 2 : continue to spend at the original rate

EAC = AC + (BAC – EV)

Assumption 3 : both CPI and SPI influence the remaining work

EAC = [AC + (BAC – EV)] /(CPI*SPI)

Monitoring process – Earned Value Analysis

Other useful forecasting indicators

Estimate to complete (ETC)
Variance at completion (VAC)
To complete performance index (TCPI)

Controlling process

Controlling

Means balancing scope, time, cost, resources and quality.

It introduces a risk factor and depends on stakeholders priority.

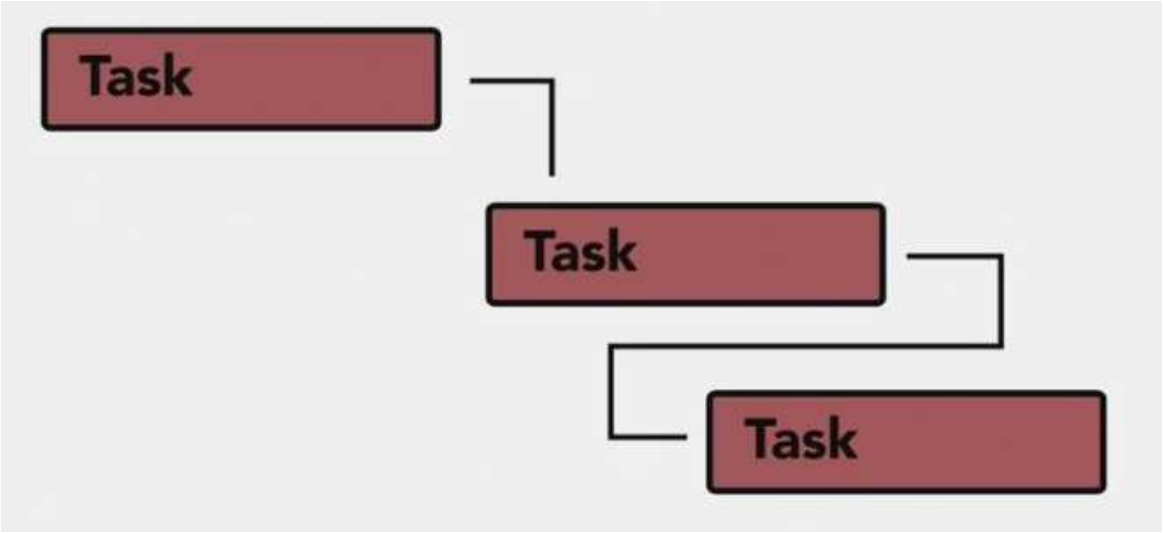
If schedule is important you can use techniques like fast-tracking and crashing

If money is a priority you can reduce costs by reducing resources or overhead costs.

If the schedule, money and resources are not negotiable reduce scope eliminating the tasks associated with it.

Controlling process - fast tracking

You push tasks to occur faster than they would, normally introducing negative lag time



Controlling – crashing

Shorten the tasks on the critical path

task name	Duration in days	crash lenght	crash cost	cost per day
task 1	10	2.5	1000	400
task 2	4	2	500	250
task 3	4	1	500	500
task 4	5	1	600	600
task 5	4	1	1000	1000

Controlling process - closing

- Ensure project acceptance
- Track project performance
- Lessons learned
- Close contracts
- Release resources