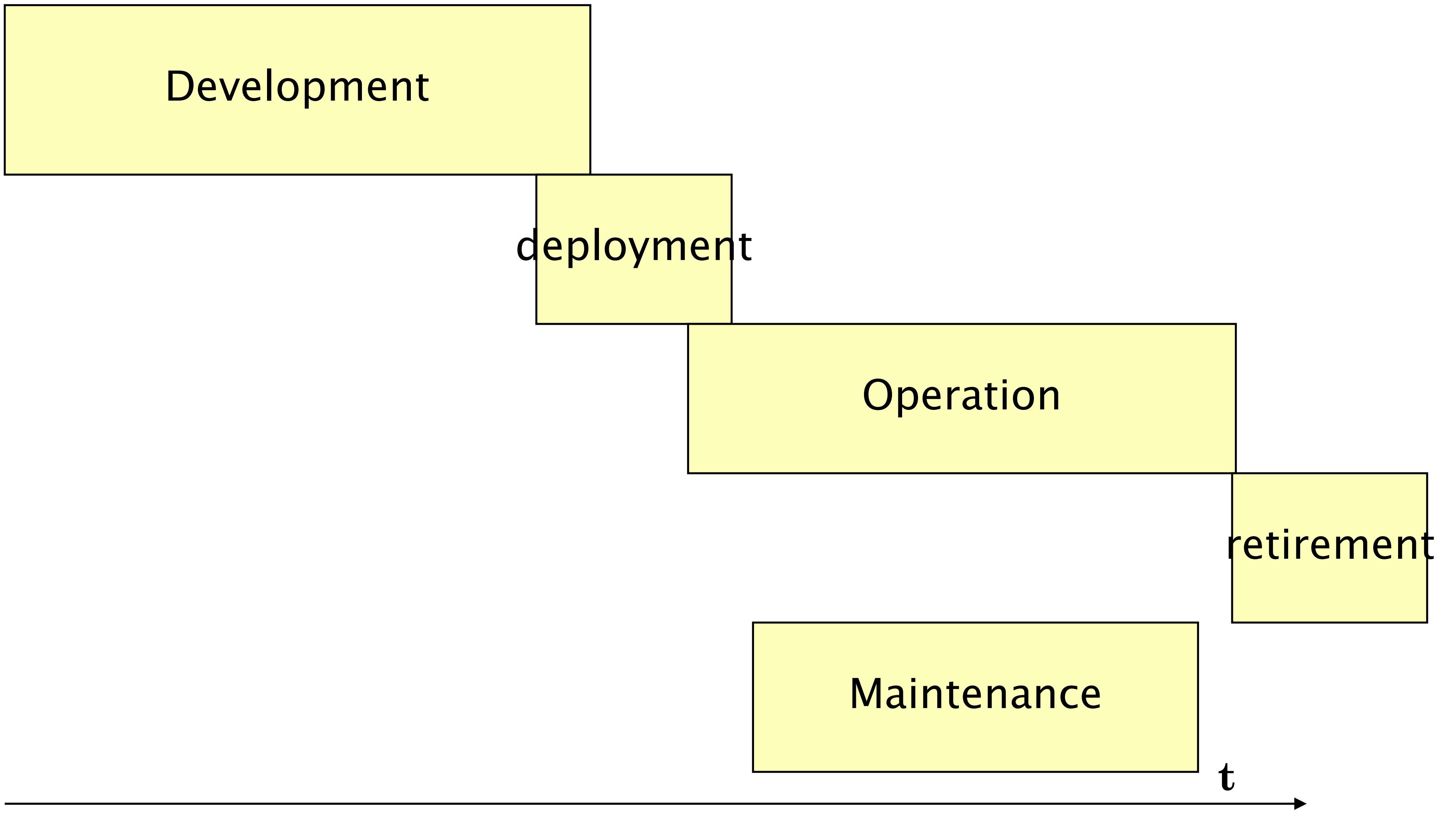
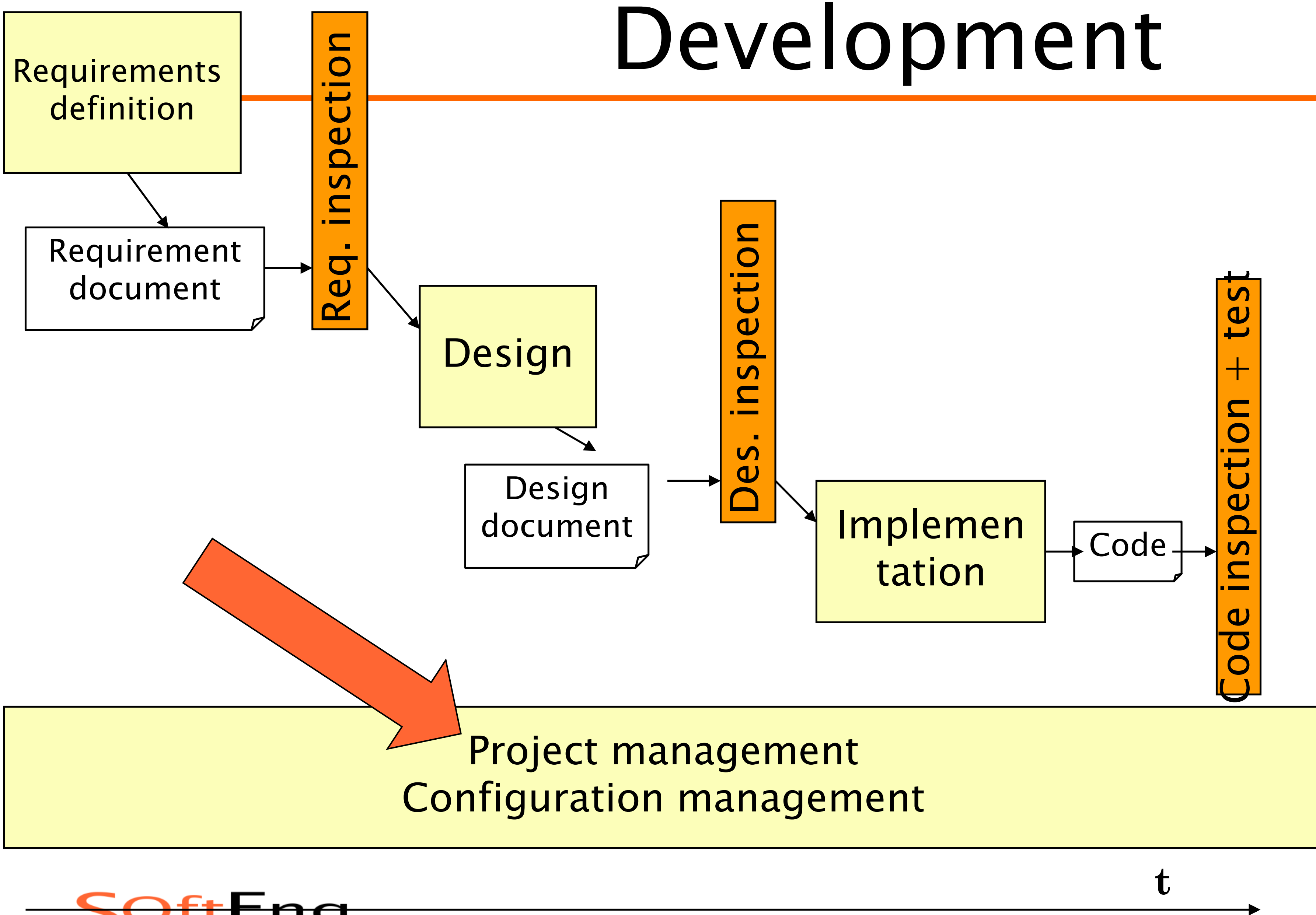
Project Management

SOftEng
http://softeng.polito.it

Main Phases


Development
deployment
Operation
retirement
Maintenance
t

Development



- Plans are worthless
- But planning is everything
    - Dwight Eisenhower

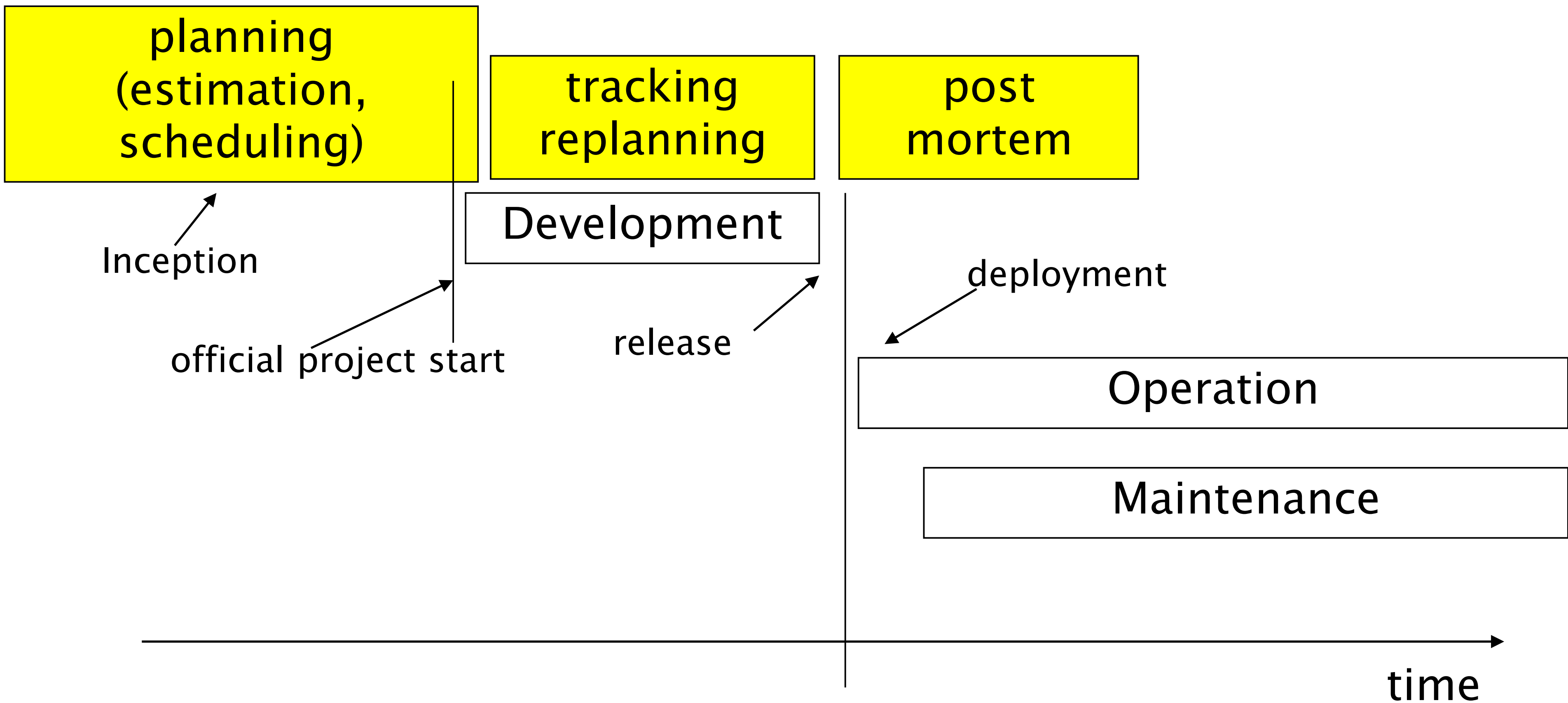
Project management

- How much?
  - Upfront: Estimation
  - During and after: tracking
- When ready?
  - Upfront: Estimation, scheduling
  - During and after: tracking
- Who does what?
  - Team organization, scheduling, work allocation

Outline

- Activities
- Concepts and techniques
  - Measures
  - Estimation
  - Scheduling
  - Tracking
  - Post Mortem

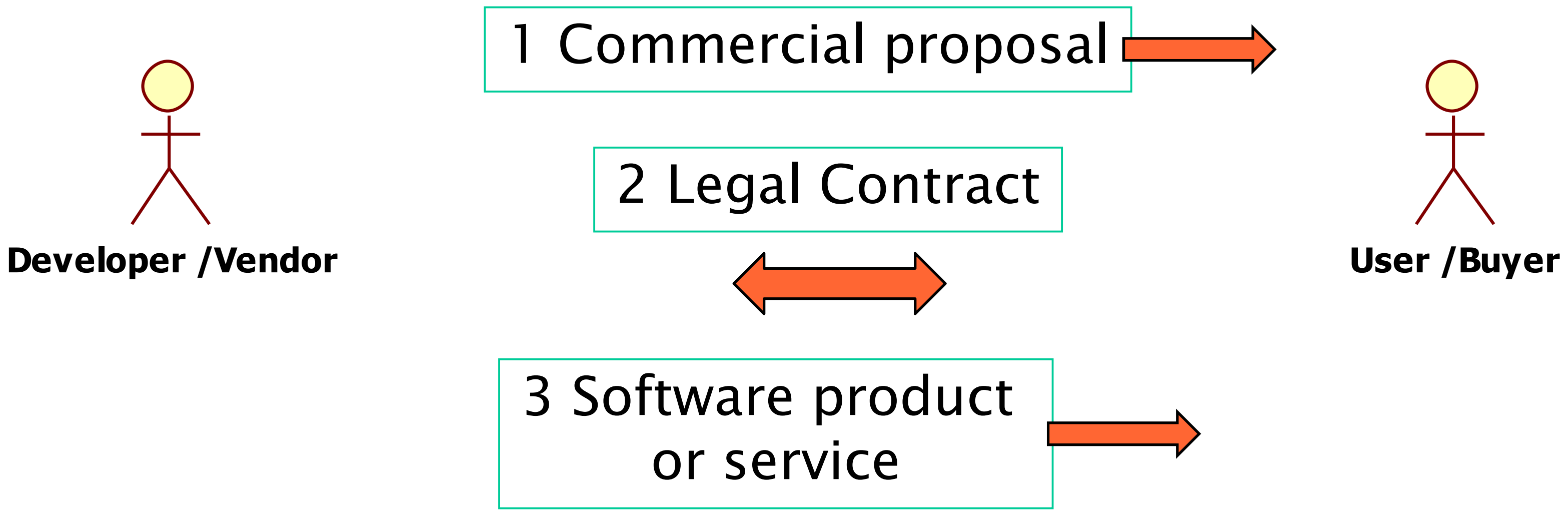
PM activities



PM – inception

- Projects do not start ‘in zero time’
- **Inception** phase
  - Initial analysis of requirements
  - Initial general architecture
  - Initial estimate of duration and cost
  - Commercial proposal

The vendor / buyer relationship



PM – inception

- Not all projects pass the inception phase
- Inception phase has a cost
  - Normally not paid to vendor
  - In very large projects may be paid, as ‘concept development’ or ‘feasibility study’

Common issues

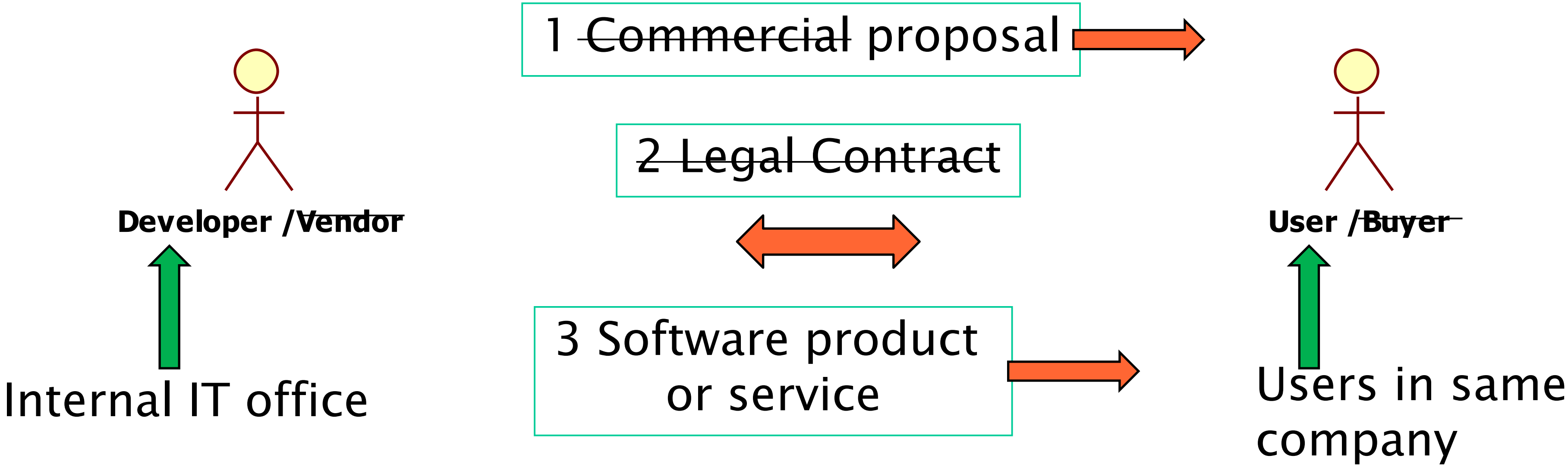
- Vendor tends to overpromise (or underestimate cost and time) to gain the contract
  - Counting on variations (and related costs / gains) to be made later
  - Counting on gains in maintenance / operation phase (buyer is typically locked in to vendor for maintenance)

Vendor buyer – variant

- The software product / service could be developed internally
  - Same organization develops and uses

  - Typically can happen in large organizations that have an internal group/ office in charge of IT services

Vendor / buyer – variant



PM – development

- Proposal signed (as technical annex to legal contract)
- Development starts

- More detailed analysis of requirements, more detailed design
- Planning: more detailed estimation of duration and cost, scheduling activities and deliveries on calendar
- deciding organizational structure
- allocating resources

PM – development

- Development proceeds further

- Every week / month:
- Tracking: track usage of resources (time and money spent – cfr time sheet), compare with plan
- Replanning: in case of (large) deviations from plan, re-plan

PM – post mortem

- Project ended
- Learn for next projects

- **Post mortem** phase
  - Analyze good / bad choices, mistakes and successes, things to repeat, things to avoid

Concepts and techniques

Concepts and Techniques

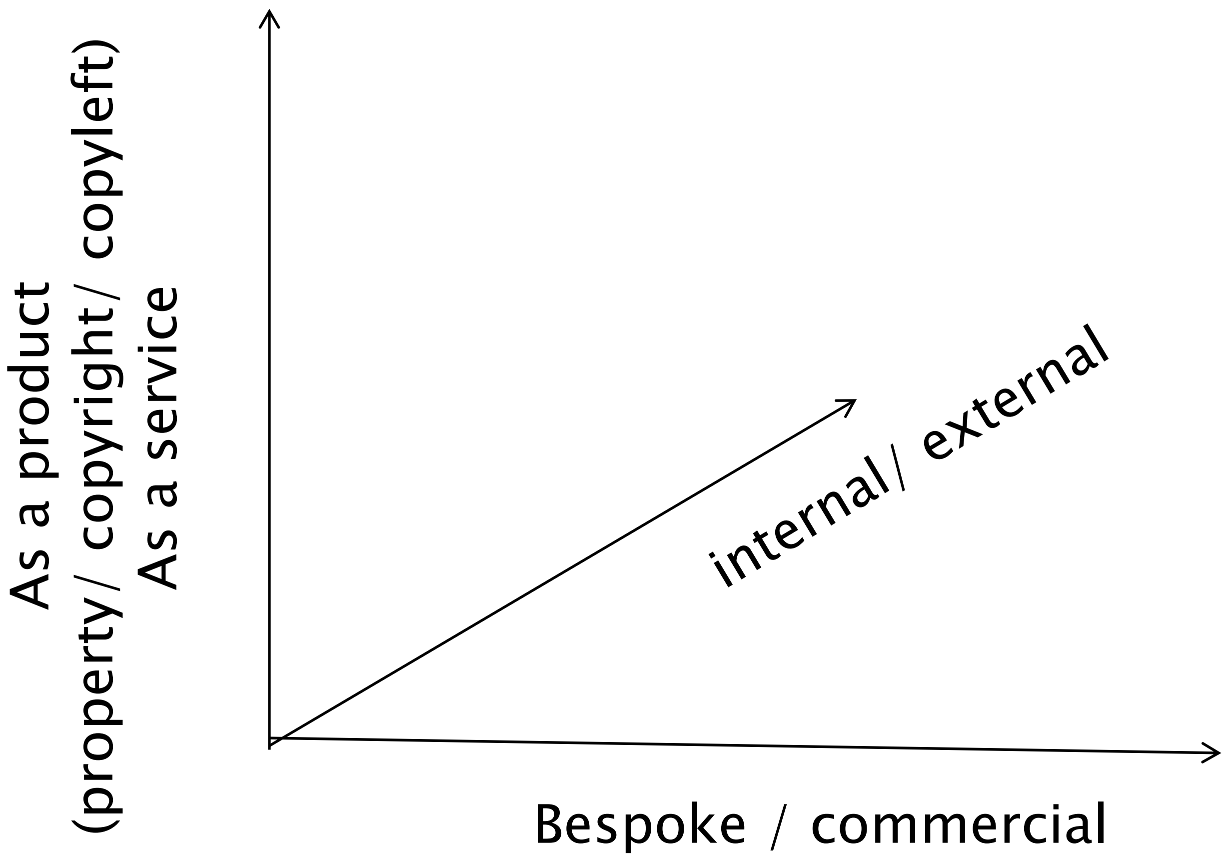
- Concepts
  - Project, program
  - Dimensions of projects
  - Resource
  - Phase, Activity
  - Milestone
  - Deliverable
- Techniques
  - Pert, Gantt, WBS, PBS

Project, program

- Project: collaborative endeavour to achieve a goal, with defined limits of time / money
  - Ex Manhattan project
- Program: management of several related projects
  - Ex Apollo program
    - Apollo 1 project, Apollo 2 project, ...

Dimensions of software projects

Dimensions


As a product
(property/ copyright/ copyleft)
As a service
internal/ external
Bespoke / commercial

Dimensions of software projects

- Number of buyers
  - One: bespoke
    - Ex, polito app
  - Many: commercial, COTS (commercial off the shelf)
    - Ex: MS Windows, LibreOffice, ..

→ Effect on requirement engineering

- users better defined vs ill defined
- Wider vs narrower functionality

Dimensions

- Product developed in the organization that uses it, or not
  - Internal (larger organizations have internal IT department that may develop software projects)
  - External

Effect on vendor buyer relationship, effect on cost

Dimensions

- As a service
    - Product is not installed on user’s machines
- Ownership
    - Product is installed on user’s machines
    - Property
        - Full ownership (Normally applies to bespoke)
    - Copyright
        - Property remains to producer
        - Copyright gives right to use
            - No further copy, no reverse engineering, no modification
    - Copyleft (opensource)
        - Right to use, copy and modify

→ Effect on requirements and design

Examples

- Polito app
  - Bespoke, internal, property
- Polito HR management
  - COTS, external, copyright
    - Developed by CINECA

Typical scenarios

1 Bespoke, external, property, *new*

- Company C needs custom software product P
- Company SW develops it
  - Inception, for requirement analysis and contract negotiation
  - Contract signature (cost + delivery date + functionality agreed)
  - Development, delivery

Typical scenarios

2 Bespoke, external, property, *maintenance*

- Company C needs maintenance work on owned software product P
- Company SW performs maintenance work
  - Similar to case 1, but typically contract is per year and involves a fixed amount of work (or dedicated staff) in the year (ex 400 p-days) and not an amount of functionality (as in scenario 1)

Typical scenarios

1-a, 2-a

(same as above, but internal)
Bespoke, internal, property

- typical of bank, insurance,
- internal IT department instead of external vendor
- No legal contract, but similar negociation of functionality / effort requested between user department and IT department

Typical scenarios

3 COTS, external, copyright

- Company SW develops and sells a mass market product. Many users (companies, individuals) buy and use it as is
  - Ex MS and Office, Windows. Oracle, SAP and ERP products.

Typical scenarios

3-bis COTS + <u>customization</u>, external

- Company SW develops and sells a mass market product. Company C buys and customizes it
  - Customization internal
  - Customization external (made by company SW or by third party company)
  - Customized part owned by C (property) or only used with copyright

Typical scenarios

4 COTS, external, as a service

- Company SW develops and sells a mass market product. Many users (companies, individuals) buy and use it as is from the cloud
  - This option is replacing scenario 3 in most cases. Many COTS products now available both as a product and as a service

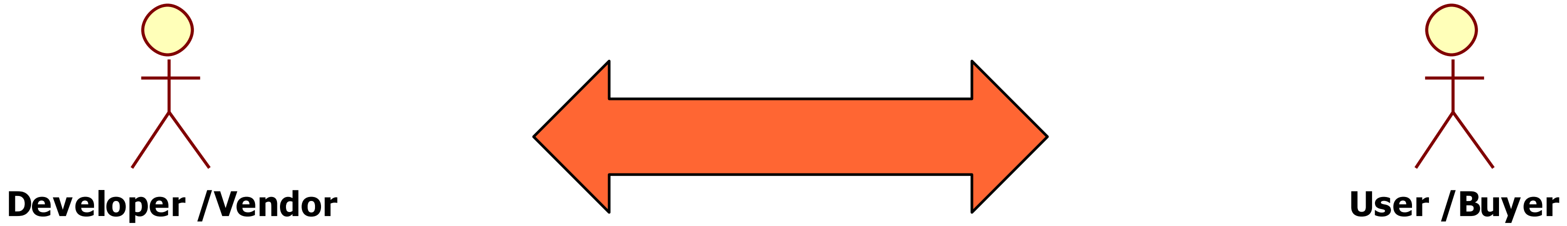
Constraints to software project

- Criticality
  - Safety critical, mission critical, other
    - Norms and laws applied, legal responsibility issues:
    - ex ISO IEC 61508 / IEC26262 for safety critical functions
- Domain
  - Aerospace, medical, automotive, industry, banking, insurance, ...
    - Norms and laws applied:
    - ex Basel 3 in finance

Constraints and costs

- Critical projects may cost way more (also10x) than non critical projects

Dimensions of buyer - vendor relationship



Contract

- Time and material
  - Contractual agreement on cost of work (time) and material
    - Ex, build a house: pay material + n person days, @M Euro / day
    - Issue: buyer may control quality in more depth
    - Issue: vendor may try to reduce productivity, final price not known in advance

Time and material, IT

- One person day (Italy, estimates)
  - Junior: 200 euro
  - Mid level: 300 euro
  - Senior, project manager: 600 euro

  - Possibly x2 in other western countries
  - Possibly /2 to /5 in lower income countries

Contract

- Fixed price
  - Contractual agreement on result and its value
    - Ex, buy a house, pay X Euro
    - Issue: price is known in advance, vendor may try to reduce quality
    - Issue: quality should be ‘perfectly’ described in technical annex to contract

Contract

- Usage fee (aka rental fee, service fee)
  - Based on agreed SLA (service level agreements)
  - Per time interval (day /month /year)
  - Per usage (#users, #calls,)

Resource

- Person
- Tool

Activity, phase

- Activity
  - Time passed by resource to perform defined, coherent task
- Phase
  - Set of activities

Milestone

- Key event/condition in the project
- with effects on subsequent activities
- ex. requirement document accepted by the customer
  - if yes then ..
  - if no then ..

Milestones example

- M0: contract signed
- M1:Requirements review passed
- M2: Design review passed
- ...

Milestones example (EZ proj)

- M1:Requirements document + UI prototype delivery
- M2: Design document delivery
- ...

Deliverable

- Product (final or intermediate) in the process
  - Cfr requirements document, prototype
- internal (for producer) or external (for customer)
- contractual value or not

Deliverables

- Ex in course project
  - Requirements document
  - GUI prototype
  - Estimation document
  - Design document
  - Code
  - Test cases (unit, integration, acceptance)
  - Time sheet

Measures

Relevant measures

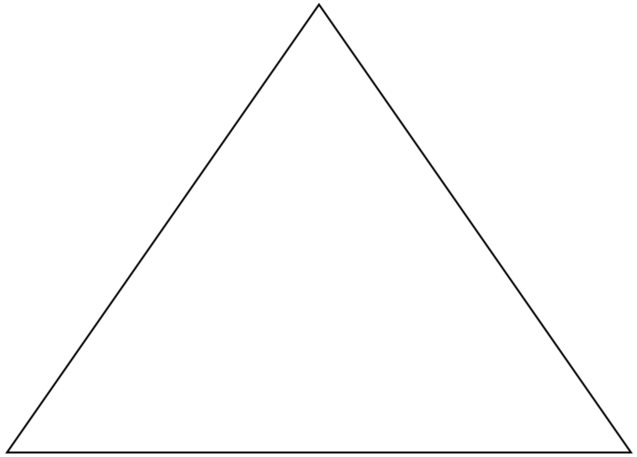
- Process measures
  - time, effort, cost
  - productivity
  - earned value
  - fault, failure, change
- Product measures
  - Functionality (FP)
  - Size
  - Price
  - Modularity
  - Other .. ilities

Variants of measure

- Estimated value: what is forecasted
  - Ex estimated duration for project X: 30 days
- Actual value: what is real
  - Ex actual duration for project X: 35 days
- Target value: what is desirable
  - Ex target duration for project X : 28 days
- Benchmark: what the others do
  - Ex benchmark duration project X: 29 days

Project Management

Software System (functions and quality)

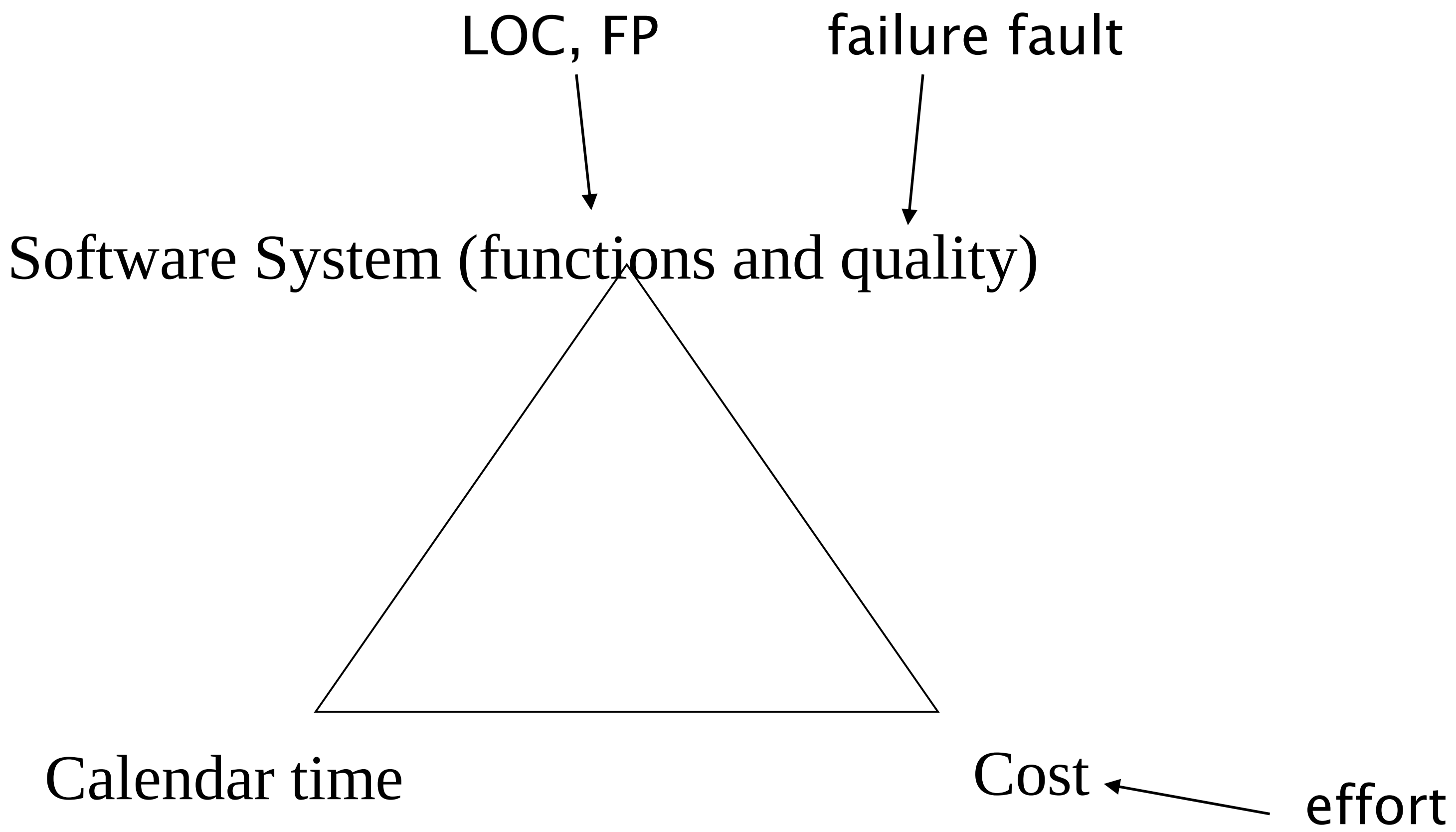


Calendar time

Cost

No notion of unpredictable events here

PM and Measures



Calendar time, or duration

- Days, weeks, months, on calendar
- Relative, from project start
  - Month1, month2, etc
  - Typically used in planning
- Absolute
  - September 12
  - Typically used in controlling
  - Remark, transition relative -> actual is not 1 to 1 (vacations, etc)

Relative to absolute

- Ex: project requires one calendar month (relative, from project start)
  - Absolute:
    - If start is August 1, and everybody is on vacation until August 31, end is September 30
    - If start is August 1, and everybody is available, end is August 31

Effort

- time taken by a resource to complete a task
- Depends on duration and on #resources employed

  duration * resource
- Measured in person hours (ieee 1045)
  - person day, person month, person year depend on national and corporation parameters

Effort

- 1 person works 6 hours → 6 ph
- 2 persons work 3 hour → 6 ph
- 6 persons work 1 hour → 6ph
- 2 persons work ½ hour → 1ph

Typical conversions

- 1 person day = 7 ph
- 1 person week = 35 ph
- 1 person month = 140 ph
- 1 person year = 1680 ph

WARNING : these conversions are NOT valid everywhere (companies and nations)

- Ex
  - Company A, Italy, person week = 38 ph
  - Company B, Italy, person week = 40 ph
  - Company C, USA, person week = 42 ph
  - Company D, France, person week = 35 ph

  - etc

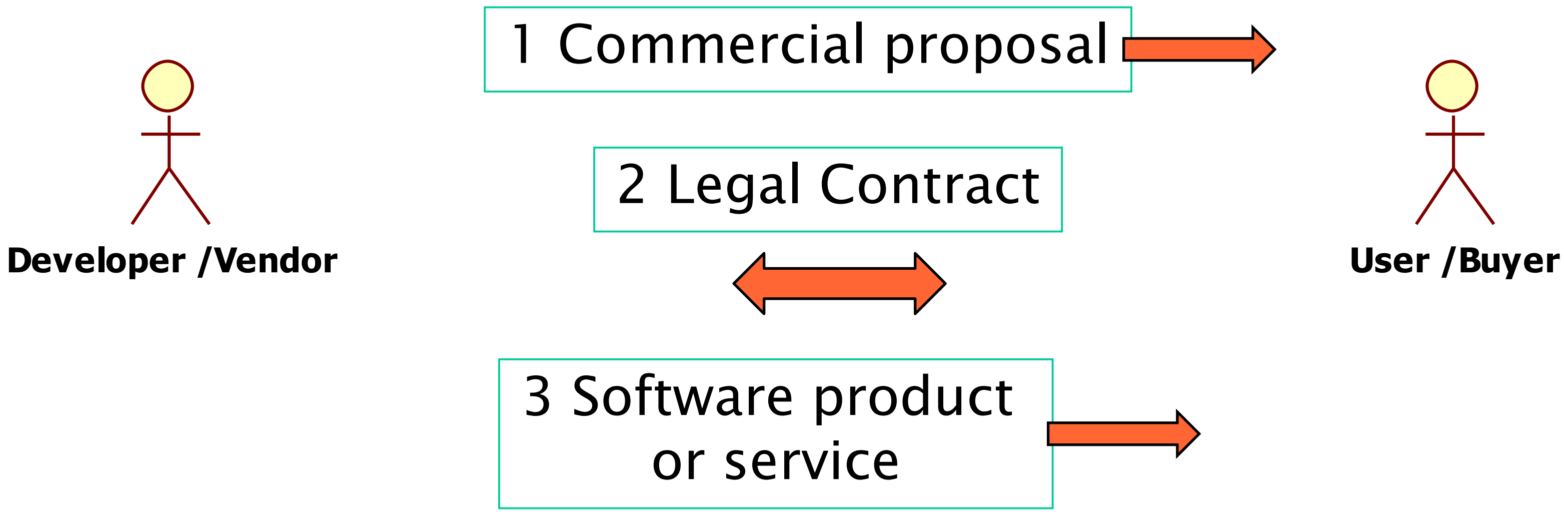
Calendar time vs. effort

- Always linked
- Mathematical link. 6 ph can last
  - 6 calendar hours if 1 person works
  - 3 calendar hours if 2 persons work in parallel
  - 1 calendar hour if 6 persons work in parallel
- Practical constraint
  - Is it feasible?
    - One woman makes a baby in 9 months
    - 9 women make a baby in one month?

Effort and cost

- Effort can be converted in cost, knowing cost per ph
  - Staff cost = person hours * cost per hour

Costs and roles



Cost – vendor

- Personnel
  - Staff
    - Person hours, salary
    - Overhead costs (office space, heating/cooling, telephone, electricity, cleaning, ..)
  - Hardware
    - Development platform, (target platform)
  - Software
    - Licenses (OS, DB, tools ..)

Cost – user

- Total Cost of Ownership (TCO)
  - Considers the complete time window involving the product
  - At least three phases
    - Before acquisition
    - Usage
    - Dismissal

Cost – user (2)

- Before acquisition
  - Costs to define requirements and select the product
    - Market analysis, feasibility studies, requirement definition, vendor / product evaluation, contract negotiation
- Acquisition
  - Acquisition cost
    - one time fee, yearly fee, usage fee
  - Acquisition cost (= price) ⇔ vendor cost + profit

Cost – user (3)

- After acquisition
  - Deployment costs
    - Install in all users machines
    - Training for users
    - Learning curve
  - Operation costs
    - Servers, network
  - Maintenance costs
    - Collection of anomalies, effect of anomalies
    - Corrective, evolutive, enhancement maintenance

Cost – user (4)

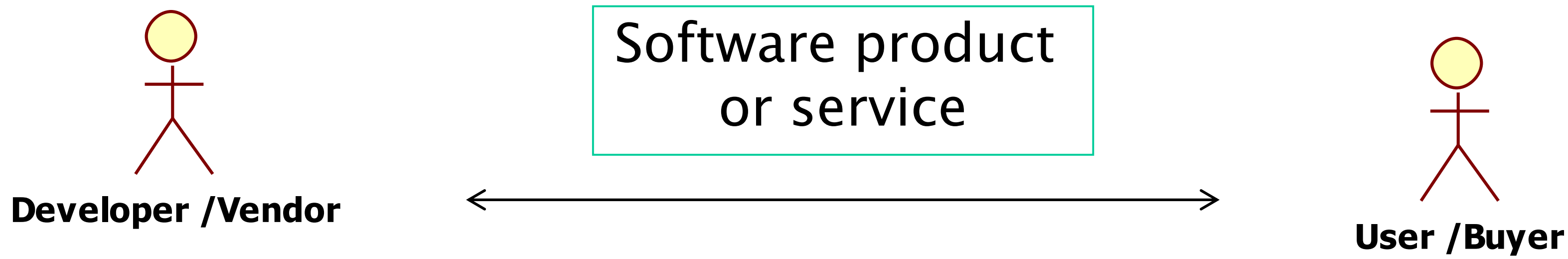
- Dismissal
  - Uninstall product
  - Back up data, data conversion ..

TCO

- The longer the time frame, the less important the acquisition cost

  - Ex, commercial airplane
  - Time frame: 20 years (50.000 hours)
  - Acquisition cost of airplane = 1/6 of TCO
    - Key TCO cost factors are fuel, crew, maintenance

Cost and price



- Cost = cost for vendor
- Price = acquisition cost for user
  - In the simplest model price = cost + margin
  - However, in the general case any relation is possible between cost and price

Software pricing factors

Factor	Description
Market opportunity	A development organisation may quote a low price because it wishes to move into a new segment of the software market. Accepting a low profit on one project may give the opportunity of more profit later. The experience gained may allow new products to be developed.
Cost estimate uncertainty	If an organisation is unsure of its cost estimate, it may increase its price by some contingency over and above its normal profit.
Contractual terms	A customer may be willing to allow the developer to retain ownership of the source code and reuse it in other projects. The price charged may then be less than if the software source code is handed over to the customer.
Requirements volatility	If the requirements are likely to change, an organisation may lower its price to win a contract. After the contract is awarded, high prices may be charged for changes to the requirements.
Financial health	Developers in financial difficulty may lower their price to gain a contract. It is better to make a small profit or break even than to go out of business.

Size

- Of source code
  - LOC (Lines of Code)
- Of documents
  - Number of pages
  - Number of words, characters, figures, tables
- Of test
  - N test cases

Size

- Of entire project
  - Function points (see later)
  - LOC
    - In this case LOCs virtually include all documents (non code) produced in the application
    - Ex. project produces 10 documents (1000 pages) and 1000 LOCs. By convention project size is 1000 LOCs

LOC

- What to count
  - w/wout comments
  - w/wout declarations
  - w/wout blank lines
- What to include or exclude
  - Libraries, calls to services etc
  - Reused components
- Comparison for different languages not meaningful
  - C vs Java? Java vs C++? C vs ASM?

Loc – what to include exclude

- Depends on the goal of counting
  - For cost estimation: count only what should be developed
  - For memory occupation: count everything
  - For price: consider whole functionality offered

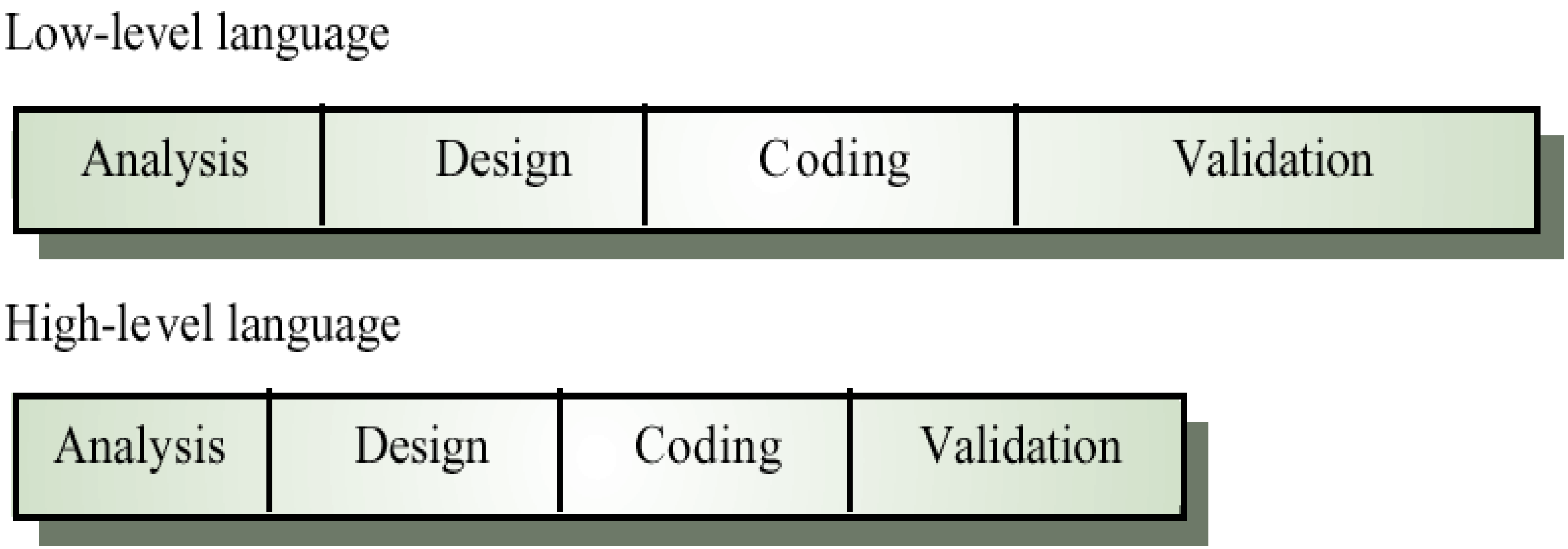
Productivity

- Output/effort
- What is output in software?
  - Size/effort = LOC / effort
    - Inherits problems of LOC
  - Functionality/effort = FP/effort
  - Object Points / effort

LOC/effort

- The lower level the language, the more productive the programmer
  - The same functionality takes more code to implement in a lower-level language than in a high-level language
- The more verbose the programmer, the higher the productivity
  - Measures of productivity based on lines of code suggest that programmers who write verbose code are more productive than programmers who write compact code

High and low level languages



Productivity paradox

	analysis	design	coding	testing	doc	Effort [p w]	Size [loc]	Productivity [loc / pw]
Low level	3 [person weeks]	5	8	10	2	28	5000	5000/28 = 178
High level	3	5	4	6	2	20	1500	1500/20 = 75

Productivity figures

- Real-time embedded systems, 40-160 LOC/P-month
- Systems programs , 150-400 LOC/P-month
- Commercial applications, 200-800 LOC/P-month

- Source: Sommerville

Productivity figures

Manufacturing	0.34 FP/person hour
Retail	0.25
Public administration	0.23
Banking	0.12
Insurance	0.12

Source: Maxwell, 1999

Factors affecting productivity

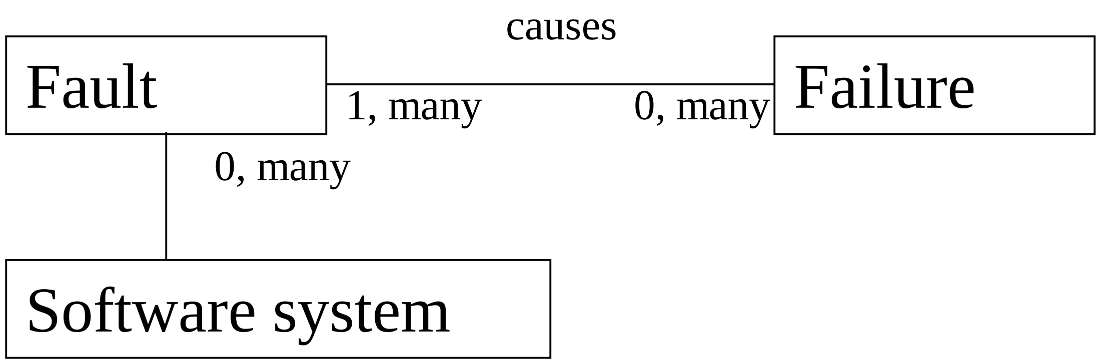
Factor	Description
Application domain experience	Knowledge of the application domain is essential for effective software development. Engineers whoalready understand a domain are likely to be the most productive.
Process quality	The development process used can have a significant effect on productivity. This is covered in Chapter 31.
Project size	The larger a project, the more time required for team communications. Less time is available for development so individual productivity is reduced.
Technology support	Good support technology such as CASE tools, supportive configuration management systems, etc. can improve productivity.
Working environment	As discussed in Chapter 28, a quiet working environment with private work areas contributes to improved productivity.

Quality and productivity

- All metrics based on size/effort are flawed because they do not take quality into account
- Productivity may generally be increased at the cost of quality
- It is not clear how productivity/quality metrics are related
- If change is constant then an approach based on counting lines of code is not meaningful

Failure vs. Fault

- Failure
  - malfunction perceived by the user
- Fault
  - defect in the system, may cause failure or not



Failure

- data to collect
  - calendar time, project time, execution time
  - effect (bad data, loss of data, ...)
  - location (product type, id)
  - gravity (human injury, economic loss, ..)
  - user profile
  - related fault(s)
- measures
  - classification, count per class
  - average time intervals

Fault

- data to collect
  - effect (related failure, if any)
  - location (product type, id)
  - type (e.g. missing req, uninitialized var, logic error, ..)
  - cause (communication, misunderstanding, clerical, ..)
  - detecting method (test, inspection, ..)
  - effort (finding and report handling)

Change

- data to collect
  - location
  - cause (related fault if corrective, adaptive, perfective)
  - effort
- measures
  - cost of failure

Fault, Failure, Change

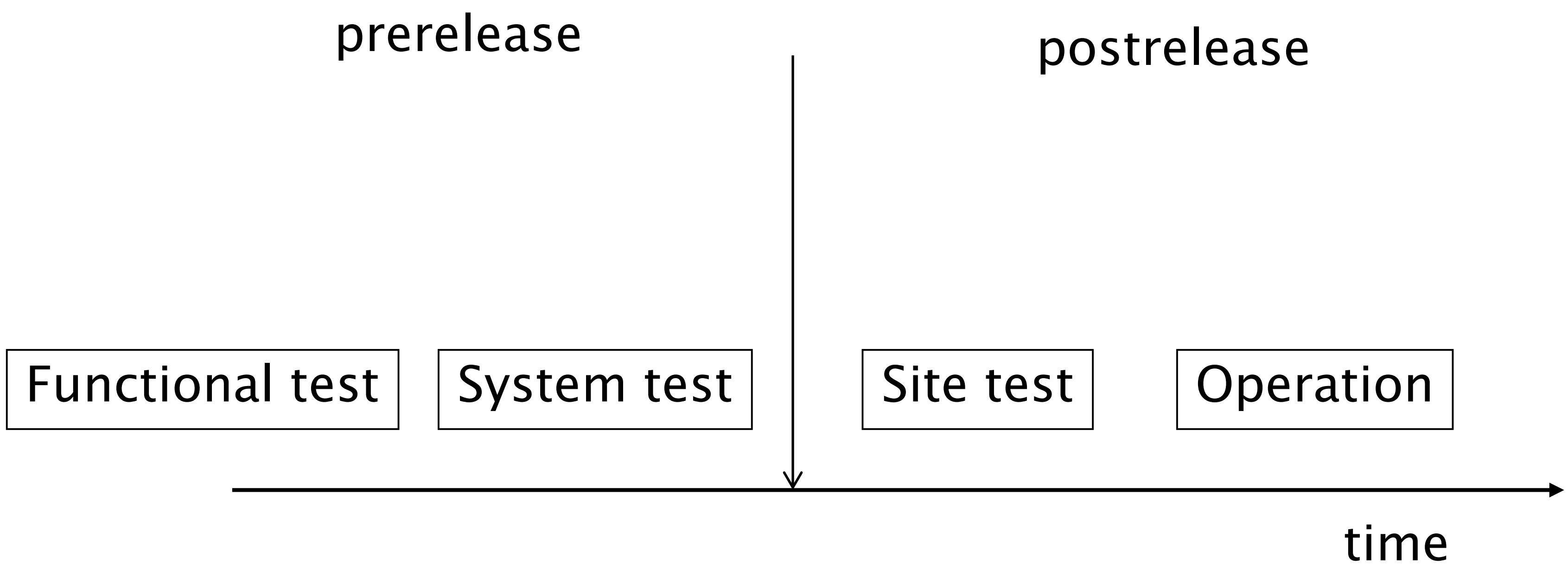
- measures
  - n open failures
  - duration/effort to close a failure
  - n failures discovered per v&v activity
  - fault/failure density
    - faults/failures per module
    - faults/failures per fp
    - faults/failures per loc
  - changes per document

Examples

- Faults per module
- Faults per phase
- Faults vs size (fault density)

Ex faults per module / per phase

- Analyzed 2 releases of large telecommunication software system (Ericsson), around 2 yrs, 200 person years per release
- 140, 246 modules, up to 6000 LOC
- 4 intervals, pre/post release



Prerelease faults

- 20% of modules cause 60% of faults
- These modules account for 30% of size

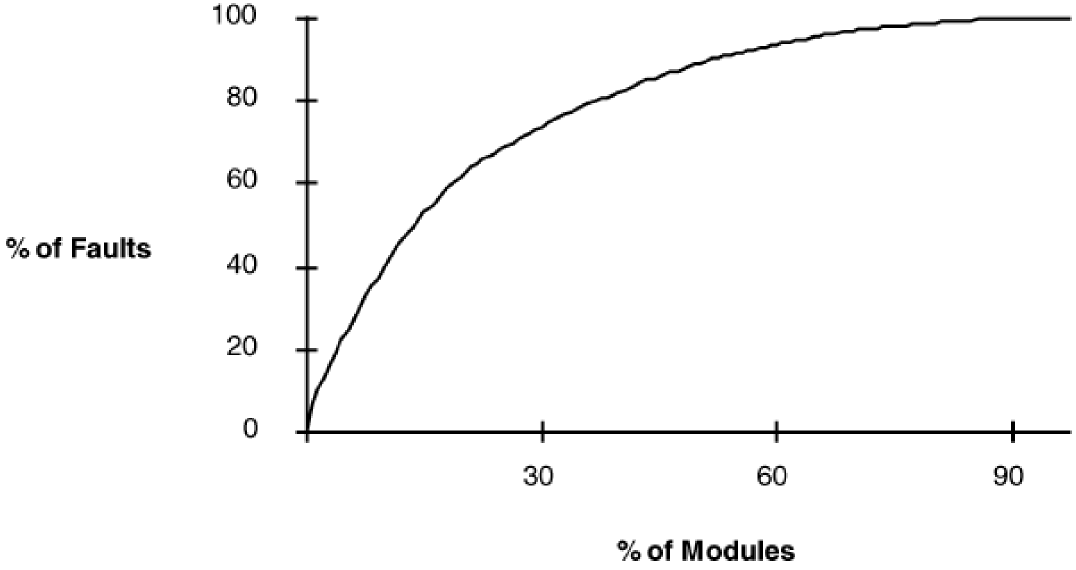


Fig. 1. Pareto diagram showing percentage of modules versus percentage of faults for *release n*.

Post release faults

- 10% of modules cause 80% of faults
- Those modules account for 10% of total size

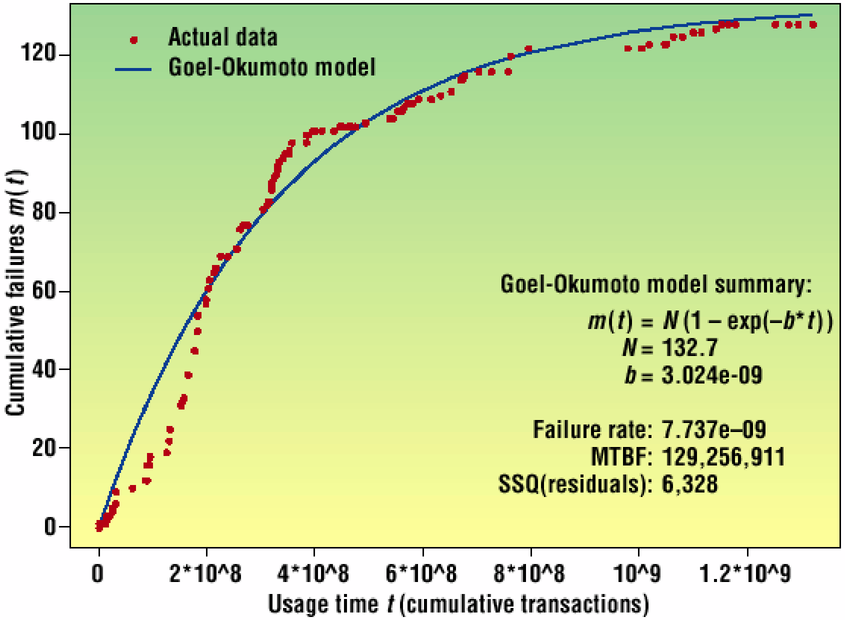
Fault densities

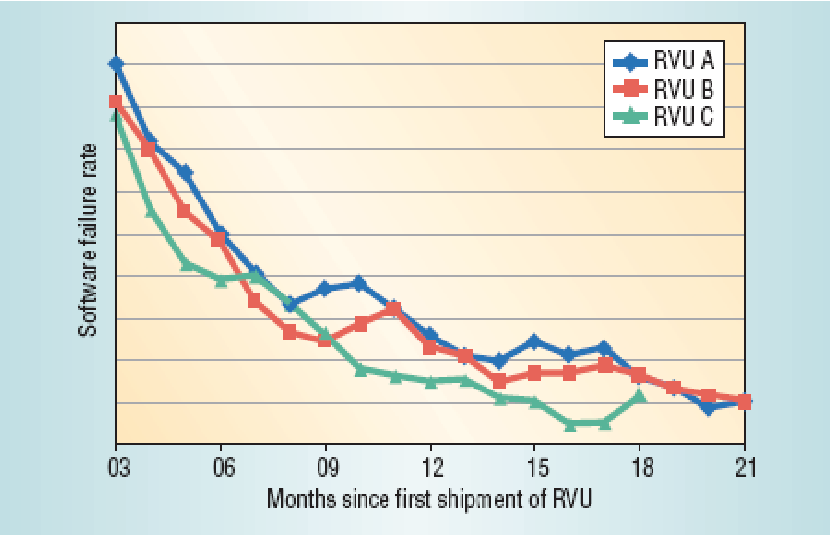
TABLE 6
Fault Densities Pre- and Postrelease for the Case Study System

	Pre-release	Post-release	All
Rel n	6.09	0.27	6.36
Rel n+1	5.97	0.63	6.60

- Benchmark
  - Good: <1fault/1KLOC
  - Bad: >10fault/1KLOC
    - Faults found in operation, 1yr after release
  - Prerelease:
    - 10-30 fault/1KLOC
  - Factor 10 between pre and post release

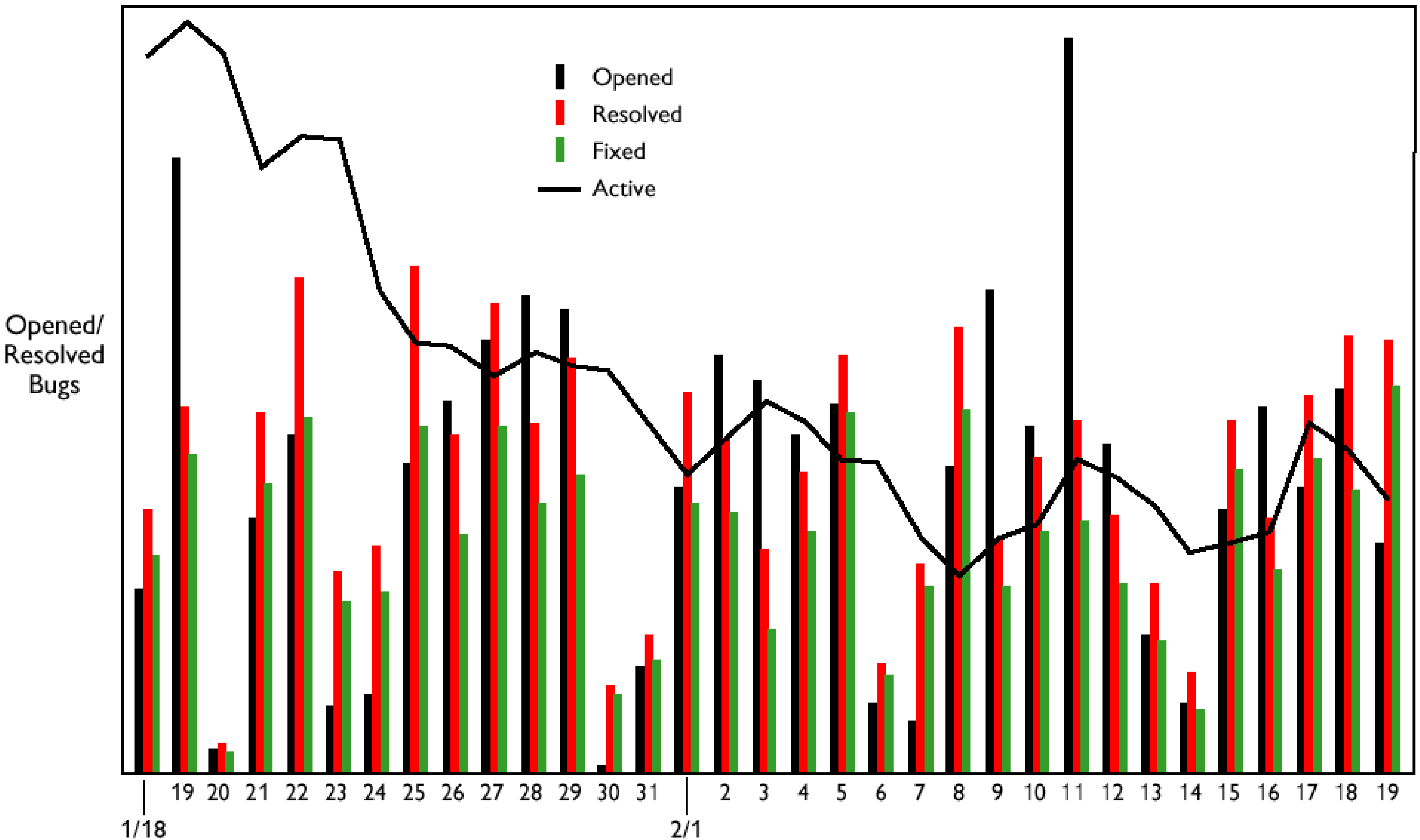
Fault rate per system





SoftEng http://softeng.polito.it

Fault rate / per status



Techniques

WBS

- Work Breakdown Structure
- Hierarchical decomposition of activities in subactivities
- no temporal relationships

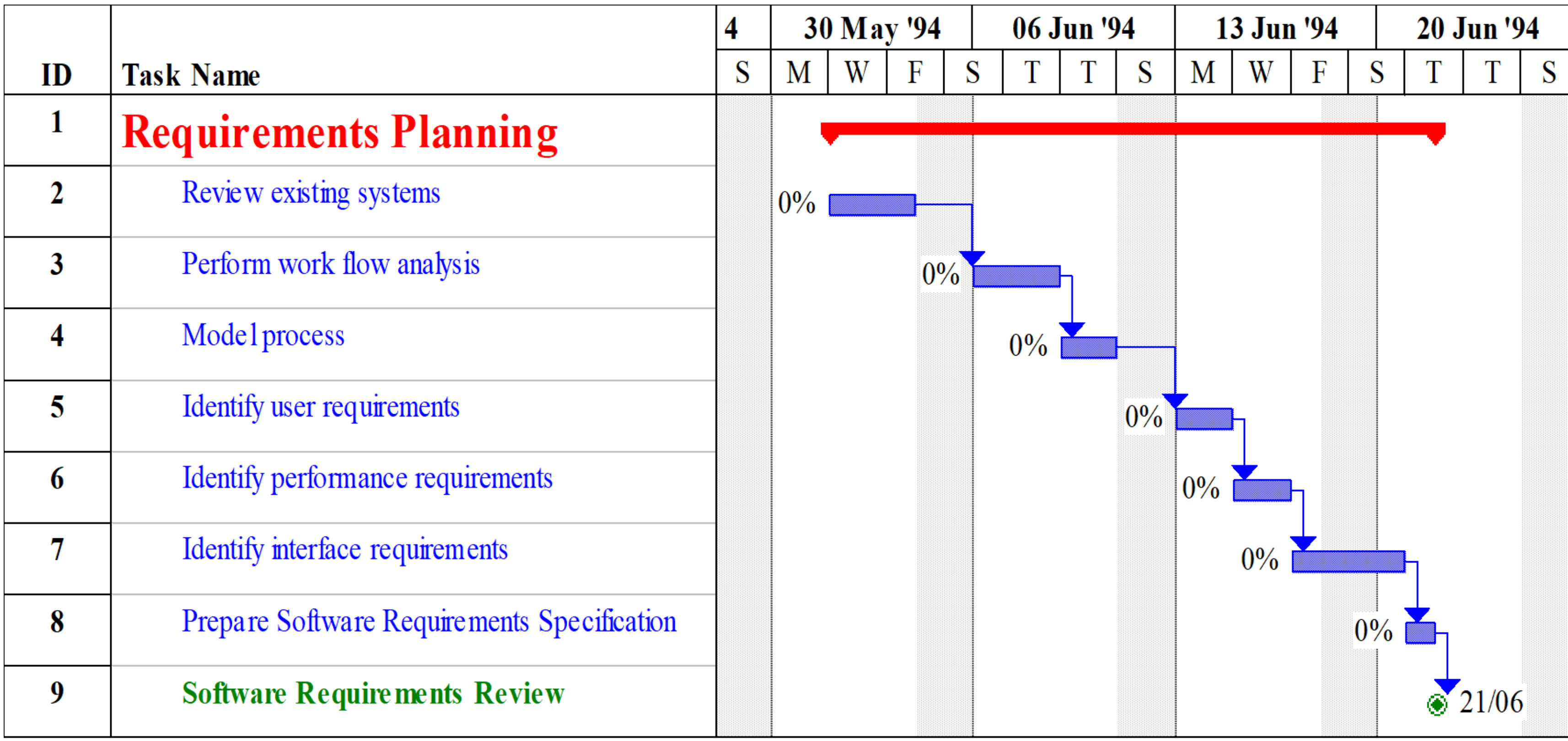
WBS example

- Requirements planning
  - Review existing systems
  - Perform work analysis
  - Model process
  - Identify user requirements
  - Identify performance requirements
  - ...
- Design
  - ...
- Implementation
  - ...

PBS

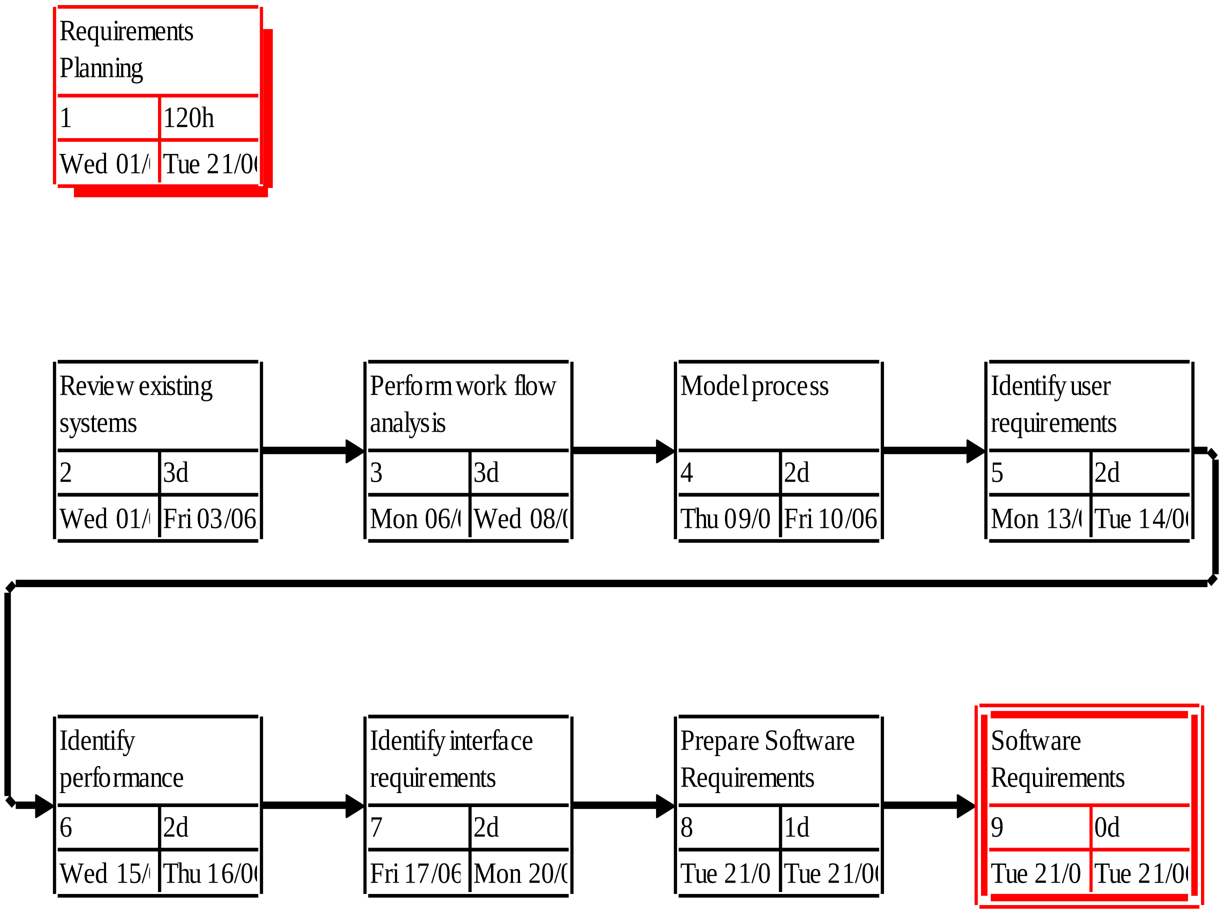
- Product Breakdown Structure
- hierarchical decomposition of product
  - Product
    - Requirement document
    - Design document
    - Module 1
      - Low level design
      - Source code
    - Module 2
      - Low level design
      - Source code
    - Testdocument

Gantt chart



ID
Task Name
4
30 May '94
06 Jun '94
13 Jun '94
20 Jun '94
S M W F S T T S M W F S T T S
1
Requirements Planning
2
Review existing systems
3
Perform work flow analysis
4
Model process
5
Identify user requirements
6
Identify performance requirements
7
Identify interface requirements
8
Prepare Software Requirements Specification
9
Software Requirements Review
0%
0%
0%
0%
0%
0%
0%
21/06

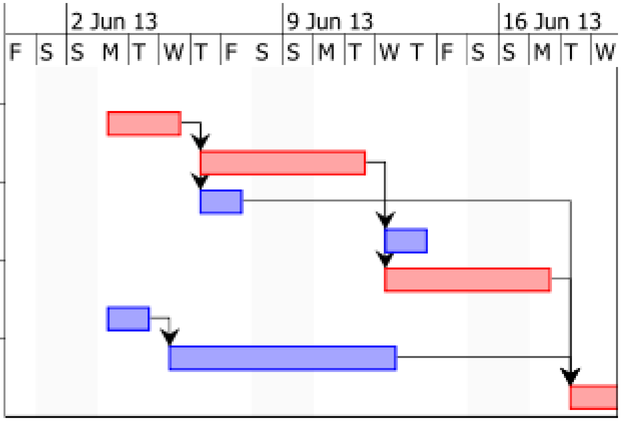
Pert



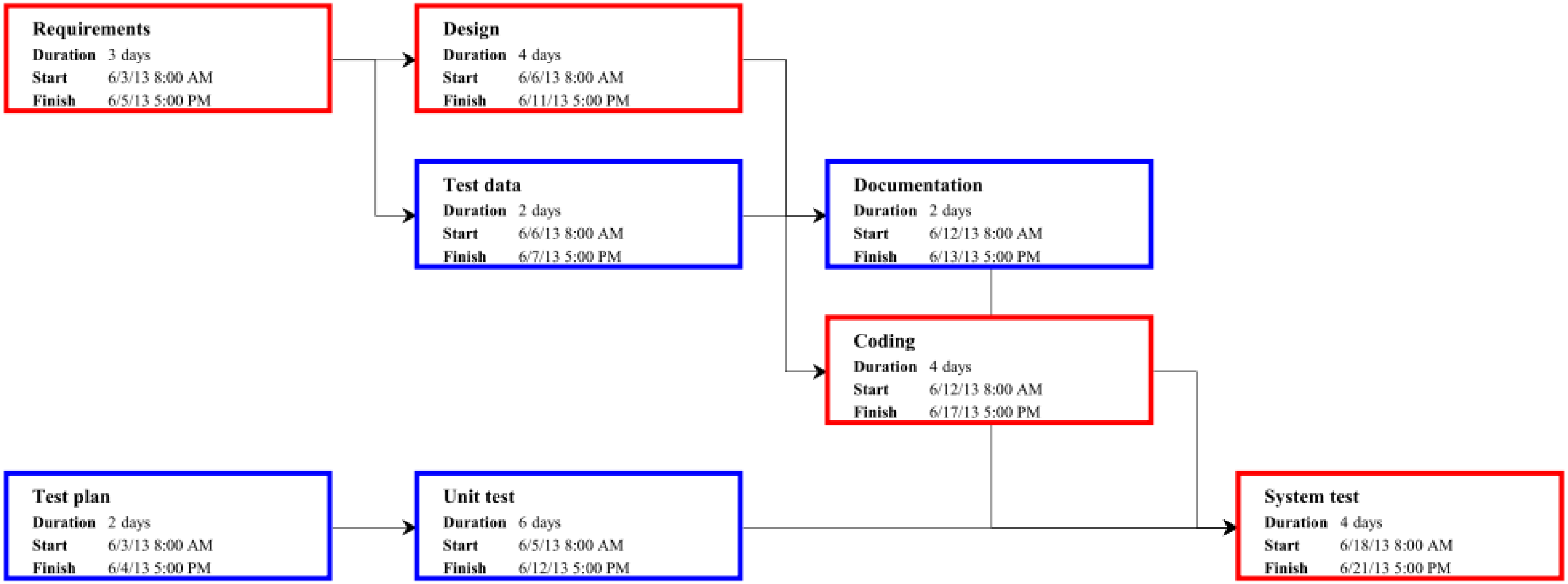
Requirements Planning
1
120h
Wed 01/
Tue 21/0
Review existing systems
2
3d
Wed 01/
Fri 03/06
Perform work flow analysis
3
3d
Mon 06/
Wed 08/
Model process
4
2d
Thu 09/0
Fri 10/06
Identify user requirements
5
2d
Mon 13/
Tue 14/0
Identify performance
6
2d
Wed 15/
Thu 16/0
Identify interface requirements
7
2d
Fri 17/06
Mon 20/
Prepare Software Requirements
8
1d
Tue 21/0
Tue 21/0
Software Requirements
9
0d
Tue 21/0
Tue 21/0

Gantt

Name	Duration	Start	Finish	Predecessors	Resource Names
Requirements	3 days	6/3...	6/5/1...		
Design	4 days	6/6...	6/11/...	1	
Test data	2 days	6/6...	6/7/1...	1	
Documentation	2 days	6/1...	6/13/...	2	
Coding	4 days	6/1...	6/17/...	2	
Test plan	2 days	6/3...	6/4/1...		
Unit test	6 days	6/5...	6/12/...	6	
System test	4 days	6/1...	6/21/...	5;3;7	



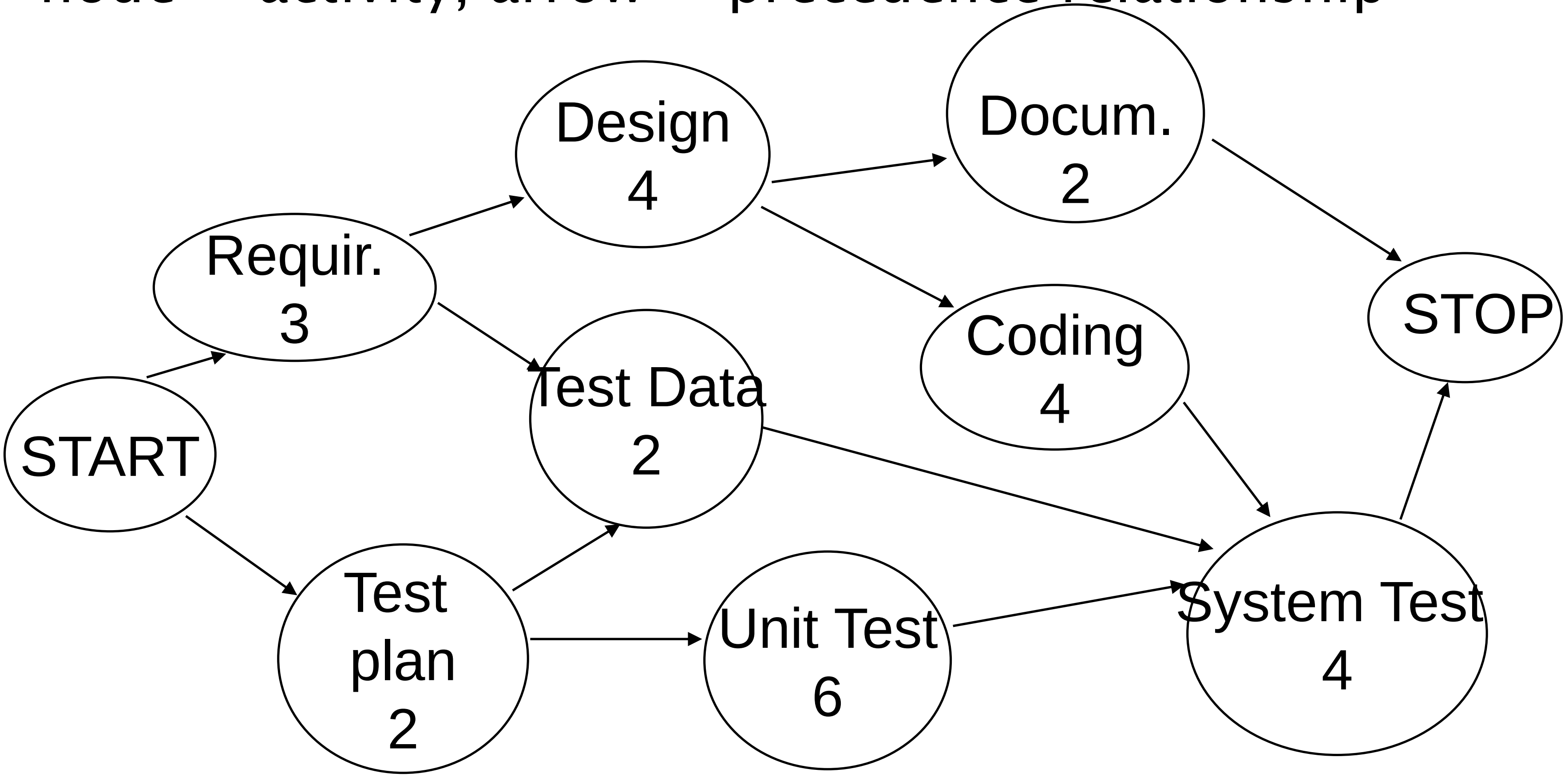
PERT


Requirements
Duration 3 days
Start 6/3/13 8:00 AM
Finish 6/5/13 5:00 PM
Design
Duration 4 days
Start 6/6/13 8:00 AM
Finish 6/11/13 5:00 PM
Test data
Duration 2 days
Start 6/6/13 8:00 AM
Finish 6/7/13 5:00 PM
Documentation
Duration 2 days
Start 6/12/13 8:00 AM
Finish 6/13/13 5:00 PM
Coding
Duration 4 days
Start 6/12/13 8:00 AM
Finish 6/17/13 5:00 PM
Test plan
Duration 2 days
Start 6/3/13 8:00 AM
Finish 6/4/13 5:00 PM
Unit test
Duration 6 days
Start 6/5/13 8:00 AM
Finish 6/12/13 5:00 PM
System test
Duration 4 days
Start 6/18/13 8:00 AM
Finish 6/21/13 5:00 PM

PERT

directed acyclic graph:

node = activity, arrow = precedence relationship



Critical path analysis

- What is shortest time to complete the project?
- What are the critical activities to complete the project in shortest time?
- Critical activities are the ones on the critical path(s)

Critical path

Path with longest duration

start end

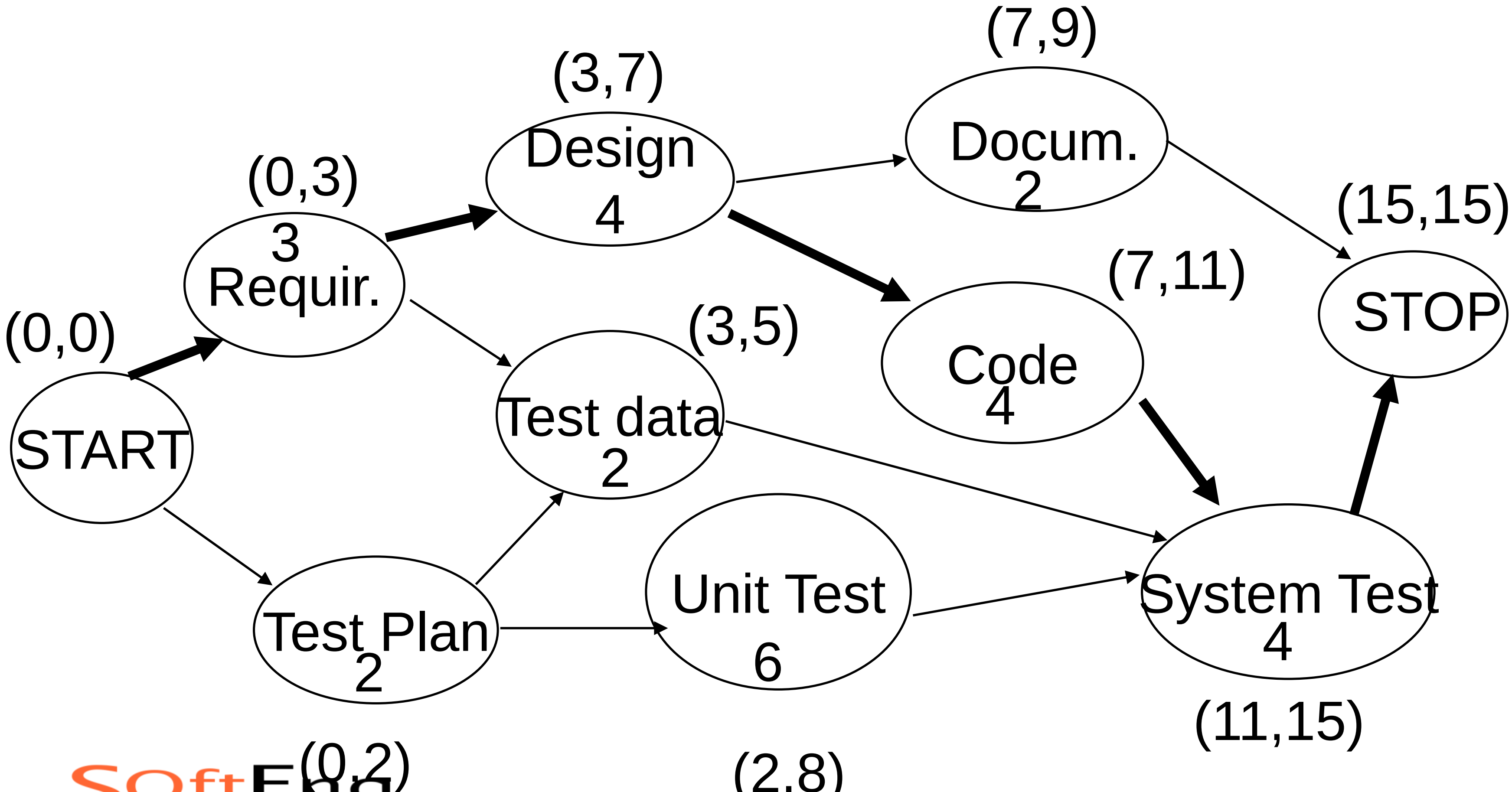
(1) START label with (0,0)

(2) For each node N whose predecessors are labeled:
SN=max {Si} Si: end time for i-th predecessor

label N with (SN, SN+duration)

(3) Repeat (2) until all nodes labeled

Example



(7,9)
(3,7)
Design
4
Docum.
2
(0,3)
3
Requir.
(15,15)
(7,11)
STOP
(0,0)
(3,5)
Code
4
START
Test data
2
Unit Test
6
System Test
4
Test Plan
2
(11,15)
(0,2)
(2,8)

Analysis

- Late start
latest time an activity can be started without changing end time of project

- Slack time
Admissible delay to complete an activity

To Compute “Slack Time”

Start from graph (S,F) from critical path analysis, for each node compute new labels (S’,F’), max start and end times

1. For STOP (S’, F’)=(S,F).
2. For each node whose successors are labeled (S’, F’) compute min S’, that becomes F’ for the node
   S’=F’-duration
   Slack Time=S’- S (or also F’- F)
3. Repeat

Managerial Implications

1. Use slack time to delay start time, or lenghten, an activity
2. If duration of activity on critical path lenghtens by X, the whole project is delayed by X
3. If only one critical path exists, reducing duration of any activity on critical path shortens duration of project.

PM approaches

- Two radically different approaches
  - Depend on process model chosen: ‘waterfall’ vs ‘iterations’
- Traditional, waterfall
  - Requirements → project plan
- Time framed, agile
  - ‘Dynamic’ requirements, iterations

Waterfall	Iterations
Define requirements	1 Define requirements (less deeply)
Estimate effort for all requirements	2 Rank requirements
Define Gantt (one iteration)	3 Pick top requirements feasible in one iteration (4 weeks)
Develop	Repeat 2,3 until finished

Ex waterfall

- 1 Define requirements
  - F1, F2, F3, F4, F5, F6, F7
- 2 Estimate
  - 120 person days
- Define Gantt
  - 3 people available
  - Roughly 40 calendar days needed
- Develop

Ex iterations

- 1 Define requirements
  - F1, F2, F3, F4, F5, F6, F7
- 2 rank
  - F7 (20pd) , F4 (25pd) , F2 (15), F5 (10), F1 (20), F3 (10), F6 (20)
- 3 Pick requirements for one iteration

  ex 3 people available x 4 weeks

  = 3x4x5 = 60 person days available

- 4 do iteration 1
  - Deliver F7, F4, F2

- Repeat ranking – requirements can change

Comparison

- In theory in both cases F1 to F7 in 40 calendar days

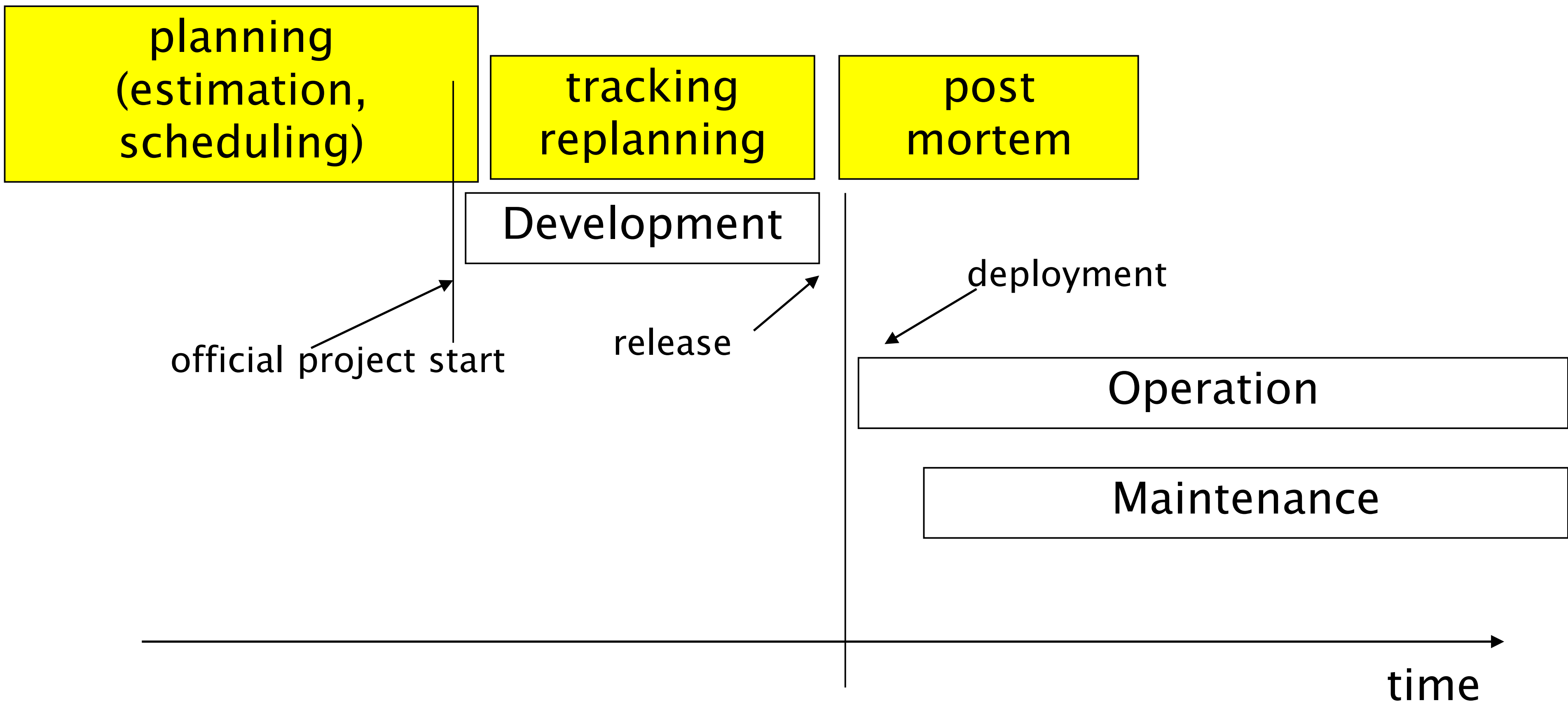
- In practice Iterations produce a different result

Comparison

- Waterfall: suitable for stable environments, larger projects, dependencies on hardware, safety critical
  - Ex automotive, aerospace
- Iterative: suitable for dynamic environments, smaller projects

- The following applies mostly to waterfall processes

The PM activities



Planning

SOftEng http://softeng.polito.it

Planning Process

- Identify activities and/or deliverables
  - PBS, WBS
  - reference models (CMM, ISO12207)
- estimate effort and cost
- define schedule (Gantt)
- analyze schedule (Pert)

Project plan

- living document
  - will be updated during tracking
- outline
  - list of deliverables, activities
  - milestones
  - Gantt
  - Pert
  - personnel organization
  - roles and responsibilities

Estimation

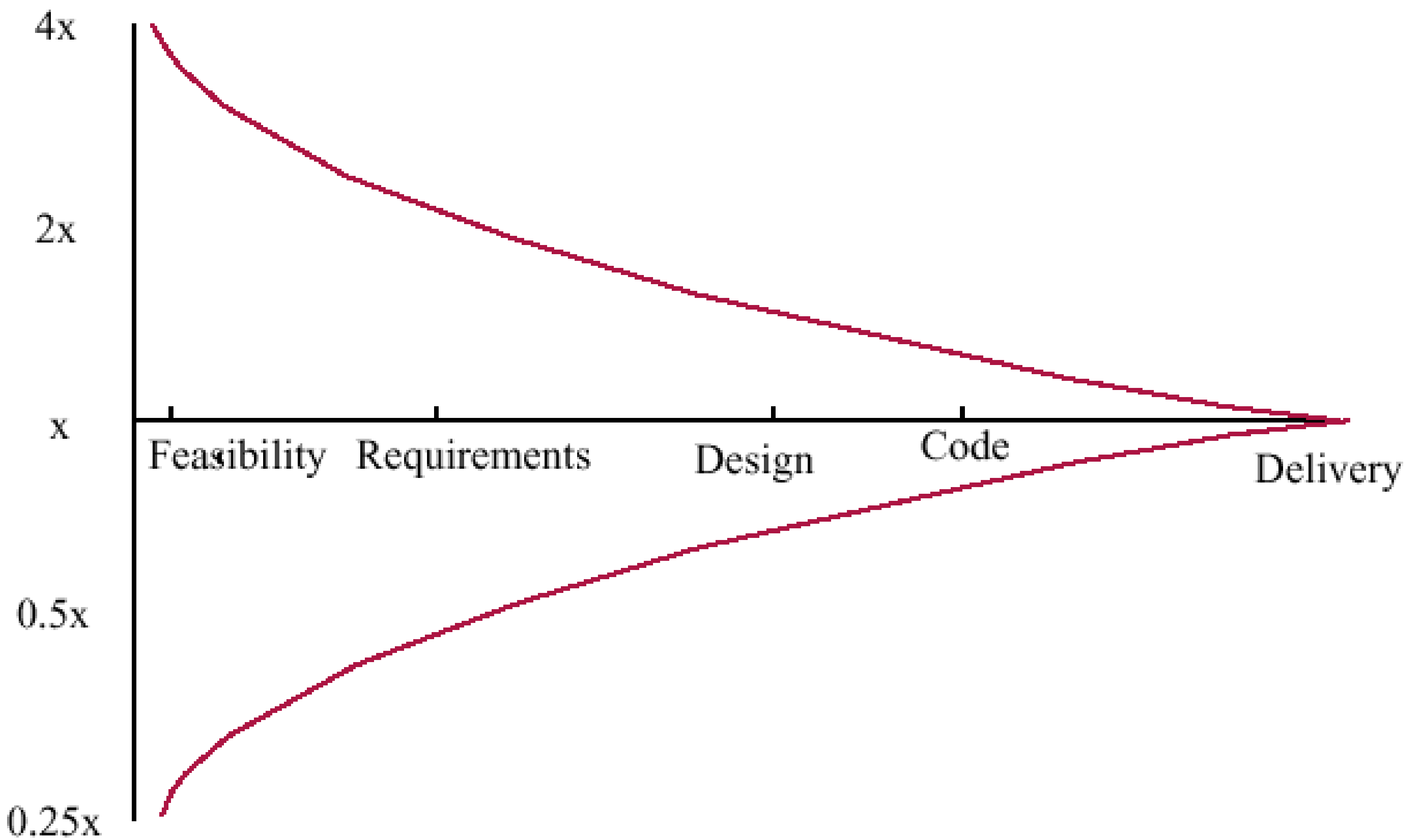
Estimation of cost and effort

- Based on analogy
  - requires experience from the past to ‘foresee’ the future
    - Experience can be qualitative (in mind of people) or quantitative (data collected from past projects)
  - the closer a project to past projects, the better the estimation

Estimation accuracy

- The cost/effort/size of a software system can only be known accurately when it is finished
- Several factors influence the final size
  - Use of COTS and components
  - Programming language
  - Distribution of system
- As the development process progresses then the estimate becomes more accurate

Estimate uncertainty



Estimation techniques

- Not suggested, but used ..
  - Parkinson’s law
  - Pricing to win

Techniques – suggested

- Based on judgment
  - Decomposition
    - By activity (WBS)
    - By products (PBS)
  - Expert judgment
  - Delphi
- Based on data from the company
  - Analogy, case based
  - Regression models
- Based on data, from outside the company
  - Cocomo, Cocomo2
  - Function points
  - Object points

Parkinson's Law

- The project costs whatever resources are available
- Advantages: No overspend
- Disadvantages: System is usually unfinished

Pricing to win

- The project costs whatever the customer has to spend on it
- Advantages: You get the contract
- Disadvantages: The probability that the customer gets the system he or she wants is small. Costs do not accurately reflect the work required

By decomposition

- By activity
  - Identify activities (WBS)
  - Estimate effort per activity
  - Aggregate (linear)

- By product
  - Identify products (PBS)
  - Estimate effort per product
  - Aggregate (linear)

- Rationale: easier to estimate smaller parts

Ex: requirements

Activity	Estimated effort (person days)
Review existing systems	5
Perform work analysis	5
Model process	1
Identify user requirements	10
Identify performance requirements	4
TOTAL	25

Expert judgement

- one (or more) experts (chosen in function of experience) propose an estimate

Delphi

- evolution of expert judgement
- structured meetings to achieve consensus in estimate
  - each participant proposes estimate (anonymous)
  - team leader publishes synthesis
    - $(a + 4m + b)/6$ (beta distribution)
    - a best – b worst – m mean
  - iterate

By analogy, case based

- A set of projects
- Each project has a number of attributes (with respective values)
  - Ex size, technology, staff experience, effort, duration, etc
- Define attributes for new project
- Find ‘near’ project(s)
  - Distance function
- Use (adapted) effort of near project

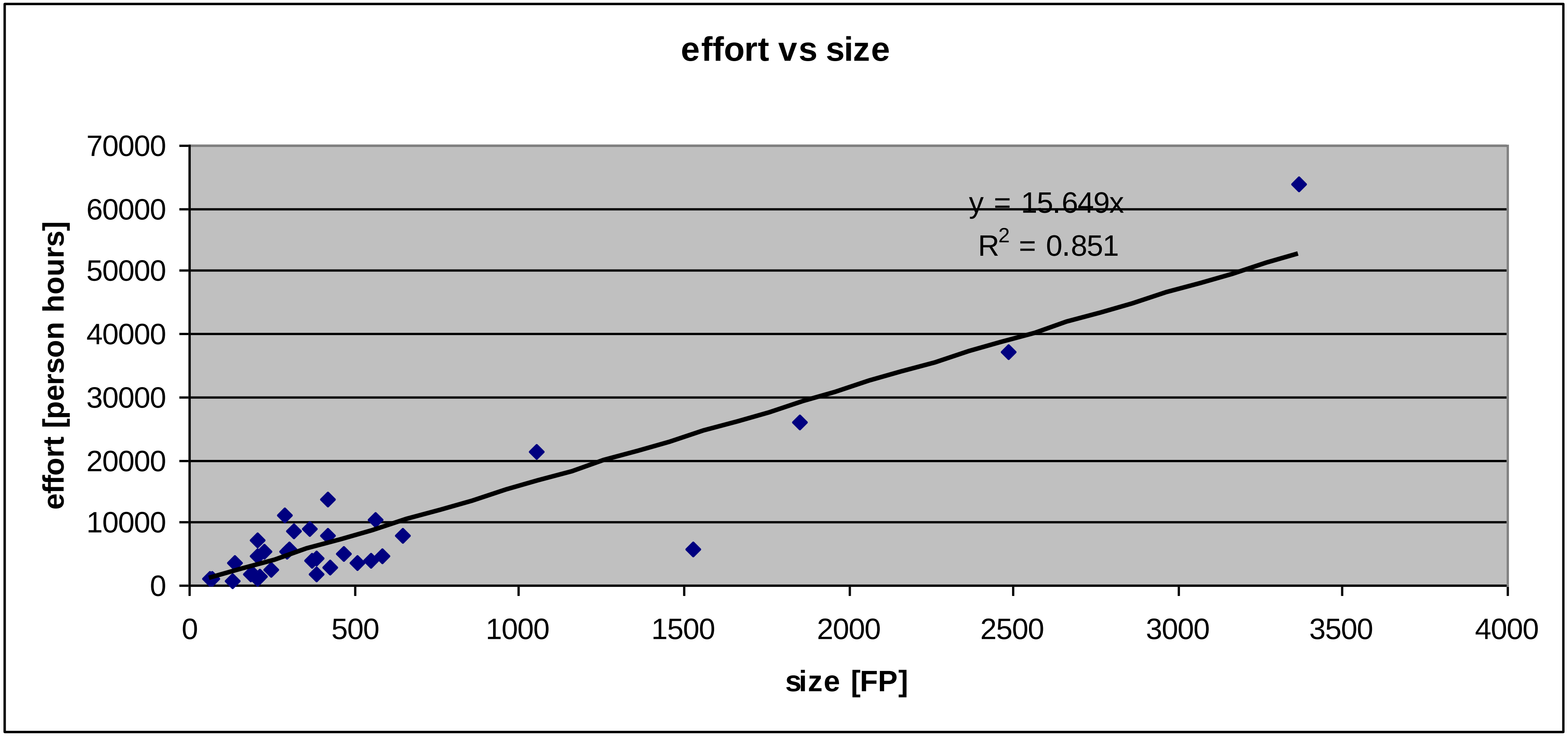
Ex.

- See file MaxwellDataSetChap1.xls
- New project
  - We estimate
    - size = 200fp, application type =transpro, telonuse = no
  - Near projects (yellow rows) have effort
    - 7320, 1520, 963, 5578
  - We estimate effort at
    - Average of effort of yellow projects= 3845

Regression models

- If the company has a data base of past projects
    - min info required: size, effort
    - See file MaxwellDataSetChap1.xls
- apply regression (linear, or else)
- Estimate productivity
- Estimate size, compute effort

Linear regression



http://softeng.polito.it

Ex.

- Using Maxwell data set, linear regression effort vs. size on all projects gives
  - Productivity = 1/15.649 fp/person hour
    0.063 fp per person hour
  - R2 = 0.85 (good model)

- Given new project
  - We estimate size =200fp
  - Estimated effort = 200*15.649 = 3773 ph

Function Points

- fp = A*EI + B*EO + C*EQ + D*EIF + E*ILF
  - EI = number of Input Item
  - EO = output item
  - EQ = Inquiry
  - EIF= External Interface File
  - ILF = Internal Logical File

- Coefficients A,B,C,D,E

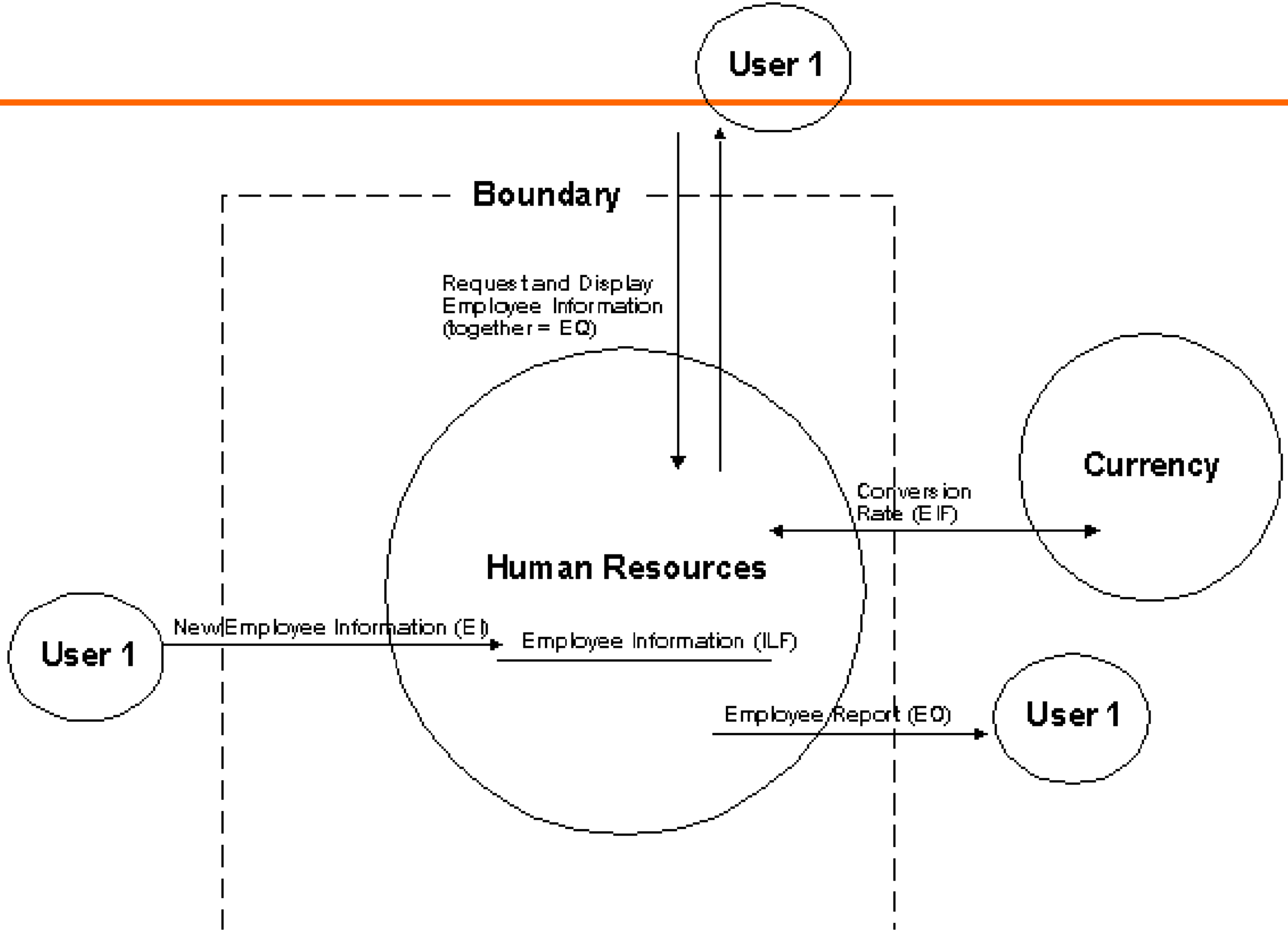
	Level of Complexity		
Component	**Simple**	**Average**	**Complex**
Input item	3	4	6
Output item	4	5	7
Inquiry	3	4	6
Master file	7	10	15
Interface	5	7	10

Function Points

- For any product, size in “function points” is given by

$$FP = 4 \times EI + 5 \times EO + 4 \times EQ + 10 \times ILF + 7 \times EIF$$

- A 3–step process.


User 1
Boundary
Request and Display
Employee Information
(together = EQ)
Currency
Conversion
Rate (EIF)
Human Resources
User 1
New Employee Information (EI)
Employee Information (ILF)
Employee Report (EO)
User 1

Function Points (2)

- 1. Classify each component of product (EI, EO, EQ, ILF, EIF) as simple, average, or complex.
  - Assign appropriate number of function points
  - Sum gives UFP (unadjusted function points)

Function Points (3)

- 2. Compute technical complexity factor (TCF)
  - Assign value from 0 (“not present”) to 5 (“strong influence throughout”) to each of 14 factors such as transaction rates, portability
  - Add 14 numbers $\Rightarrow$ total degree of influence (DI)

    $$TCF = 0.65 + 0.01 \times DI$$
  - Technical complexity factor (TCF) lies between 0.65 and 1.35

1. Data communication
2. Distributed data processing
3. Performance criteria
4. Heavily utilized hardware
5. High transaction rates
6. Online data entry
7. End-user efficiency
8. Online updating
9. Complex computations
10. Reusability
11. Ease of installation
12. Ease of operation
13. Portability
14. Maintainability

Function Points (4)

- 3. Number of function points (FP) given by

$$FP = UFP \times TCF$$

Function Points

- suitable for MIS
  - use of adjustment factors delicate
  - FP expert should do estimate
    - long, expensive
- conversion tables FP – LOC
  - Cobol 110
  - C 128-162
  - C++ 53-66
  - Java 53-62
- conversion tables FP – effort
  - www.ifpug.org

FP

- Advantage
  - Independent of technology
  - Independent of programmer
  - Well established and standardized
- Downside
  - Counting long and expensive
  - Transaction system oriented (no real time, no embedded systems)

FP vs. LOCS

	FP	LOCs
Depend on prog language	N	Y
Depend on programmer	N	Y
easy to compute	N, must be done by trained person	Y, tool based (after end of project)
Applicable to all systems	N, transaction oriented	Y

FP as unit of exchange

- Company A bids for FP
  - Buy 10000 FP, how much? (bid)
  - providers answer, x Euro per FP
- A selects provider
  - lowest cost and other factors
- End of year, redo counting
  - 10123 FP actually delivered
  - A pays

Reminder

- Measures of size
  - FP, LOC
- Both can be computed
  - Before a project start (estimated size)
  - After a project ends (actual size)
- Both can be used to
  - Characterize productivity
    - FP/effort, LOC/effort
  - Characterize application portfolio
    - FP or LOC owned and operated by a company

Function points

- IFPUG
  - FP Counting Guide
  - Exams/ certified counters
- GUFPI
- (CNIPA)

Sw project Data sets

- Company specific
  - When exists
  - Maxwell, Applied statistics for software managers, Prentice Hall
- Public
  - Knowledge plan (Caper Jones)
  - Software productivity research
  - ISBSG, Int. software benchmarking standards group, www.isbsg.org

Scheduling

Project duration

- As well as effort estimation, calendar time must be estimated, and staff allocated
- Scheduling can be done on Gantt/Pert
- COCOMO2 gives also an estimate of calendar time
  - Independent of staffing

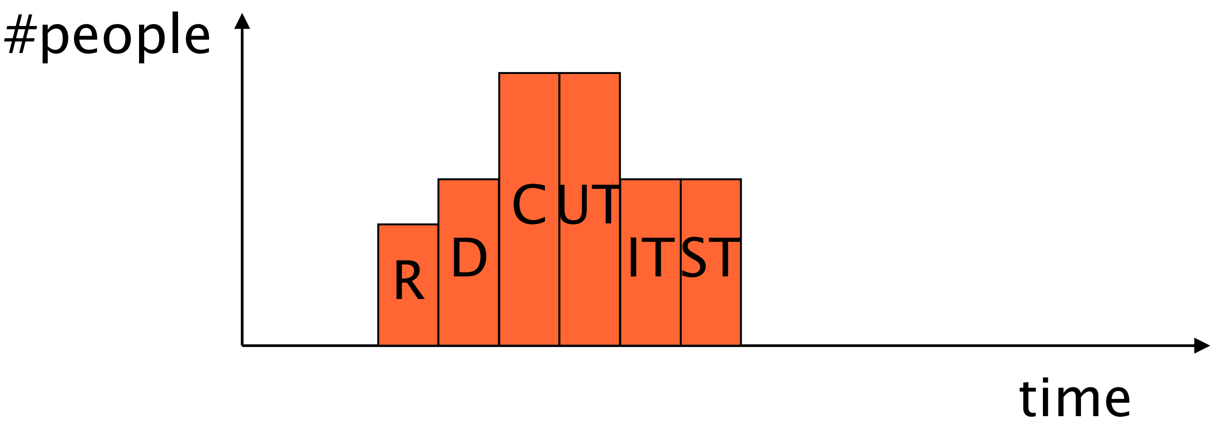
- Calendar time can be estimated using a COCOMO 2 formula
  - $TDEV = 3 \times (PM)^{(0.33+0.2*(B-1.01))}$
  - PM is the effort computation and B is the exponent computed as discussed above (B is 1 for the early prototyping model). This computation predicts the nominal schedule for the project

Staffing requirements

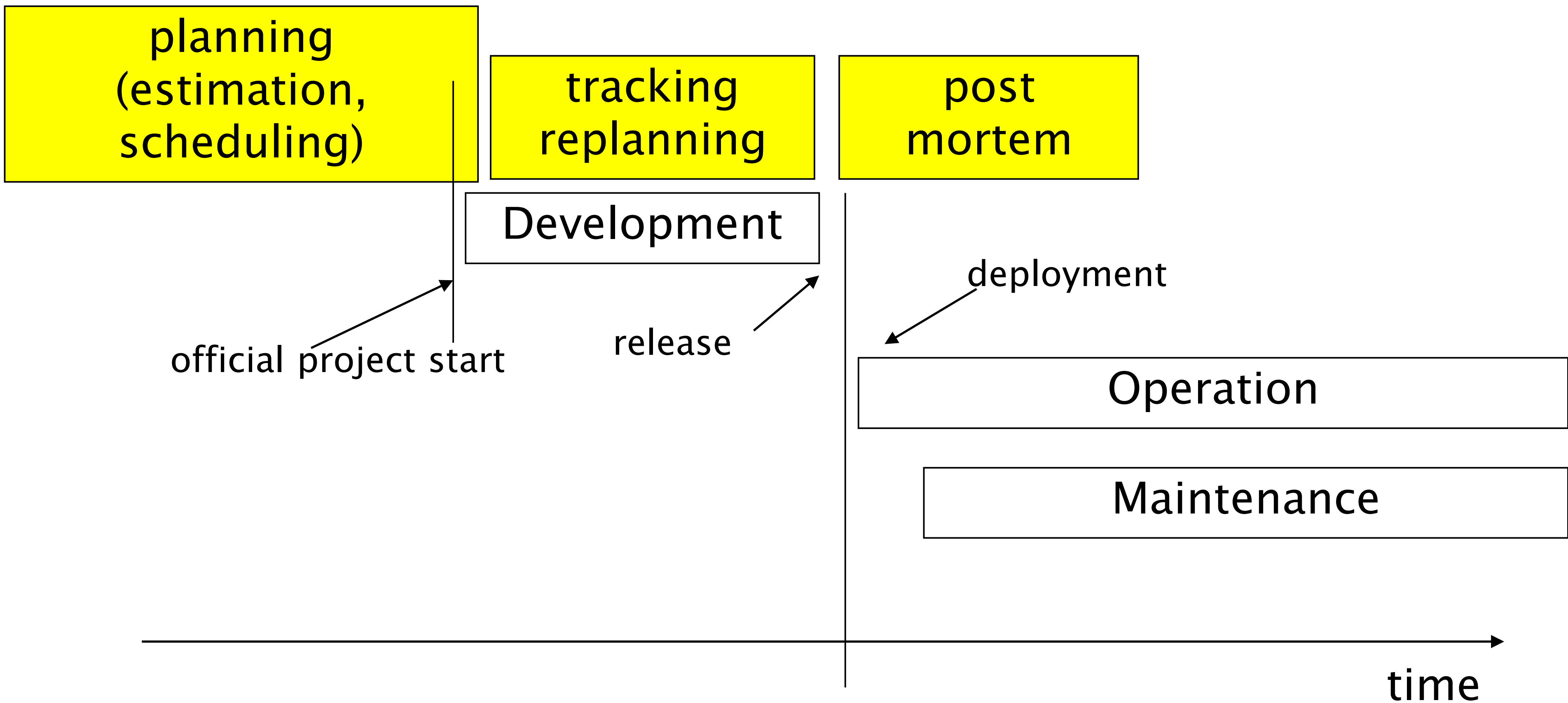
- Staff required can't be computed by dividing the development time by the required schedule
- The number of people working on a project varies depending on the phase of the project
- The more people who work on the project, the more total effort is usually required
- A very rapid build-up of people often correlates with schedule slippage

Staffing profile

- Number of people working on the project vs. time
- Typically has a bell shape
  - duration of project is constrained by staffing profile + total effort estimated



The PM activities



Tracking

Tracking process

- project has started – how to know status of project?
- collect project data, define actual status
- compare estimated – actual
  - Estimated Gantt is the roadmap for project
- if deviations, do corrective actions
  - change personnel, change activities, change deliverables, ...
  - re-plan, update Gantt and PERT

Time sheet

- Basic tool to collect <u>actual effort </u>spent by employees
  - Per day, Per project, per activity in project

- Allows to collect
  - Actual effort spent per project
- To track project status
- To compute cost of project

Time sheet -ex1

Employee name: John Smith – week 13-17 apr 2019					
day	Project X	Project Y	Sick leave	Vacation leave	Training
13 apr	4	4			
14 apr				8	
15 apr		8			
16 apr		4			4
17 apr					
TOT	4	16		8	4

Time sheet -ex2

Employee name: John Smith – week 13-17 apr 2019							
day	Project X – req	Project X – design	Project X – coding	Project X – testing	Sick leave	Vacation leave	Training
13 apr	2	2	4				
14 apr						8	
15 apr			4	4			
16 apr		2					4
17 apr							
TOT	**2**	**4**	**8**	**4**		**8**	**4**

EZGas timesheet

Week	requirement engineering	design	coding	unit testing	integration testing	acceptance testing	management	git maven
apr 13 – 19								
apr 20 – 27								
apr 28 – 3								
may 4 – 10								
may 11 – 17								

Project status

- Option1
  - Effort spent
- Option2
  - Effort spent + activities closed
- Option3
  - Earned value

Effort spent

- Collect effort spent, compare with estimated
  - Ex, spent 10, estimated 100, we are done 10%
- Big flaw, confounds input measure (effort spent) with output measure (completion)

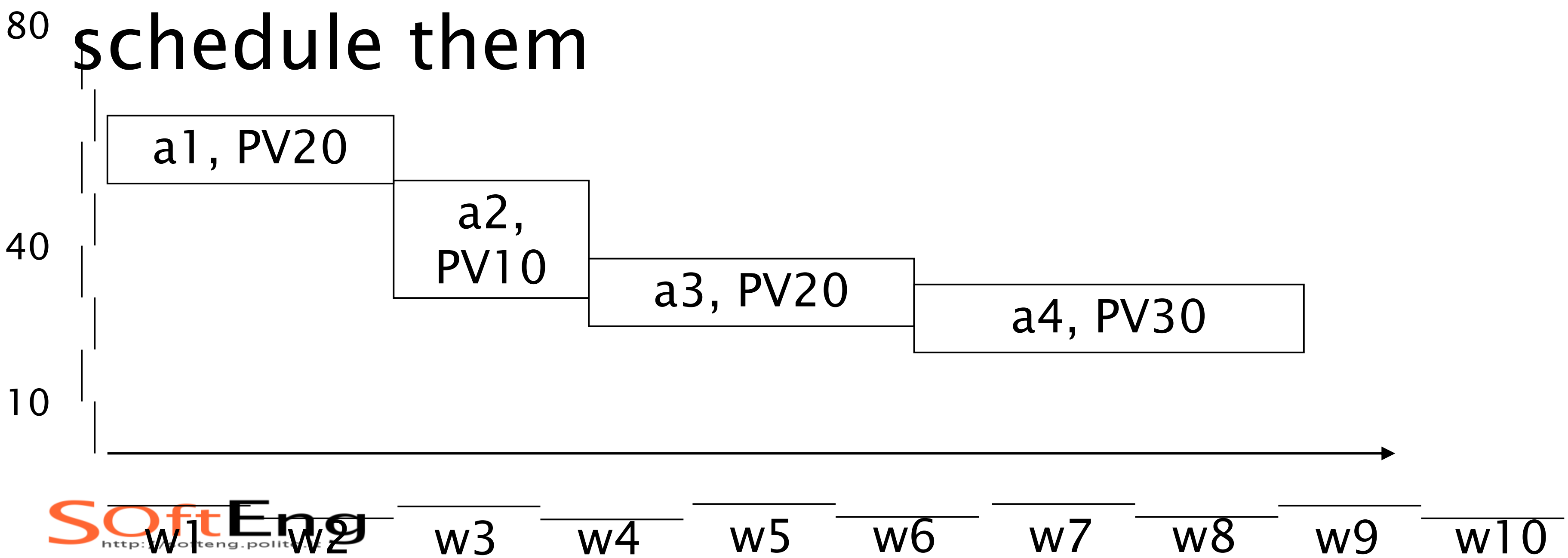
The first **90**% of a project takes **90**% of the time, the last 10% takes the other **90**% of the time
[Murphy's law]

Activities closed

- How to define when activity is closed?
  - All effort planned for activity is spent
    - Same problem, confounds input with output
  - Define quality gate, level to achieve
    - Ex, requirements: inspection meeting, majority of participants judges document is goodenough
    - Ex, unit testing: coverage 95% of nodes, and all tests pass

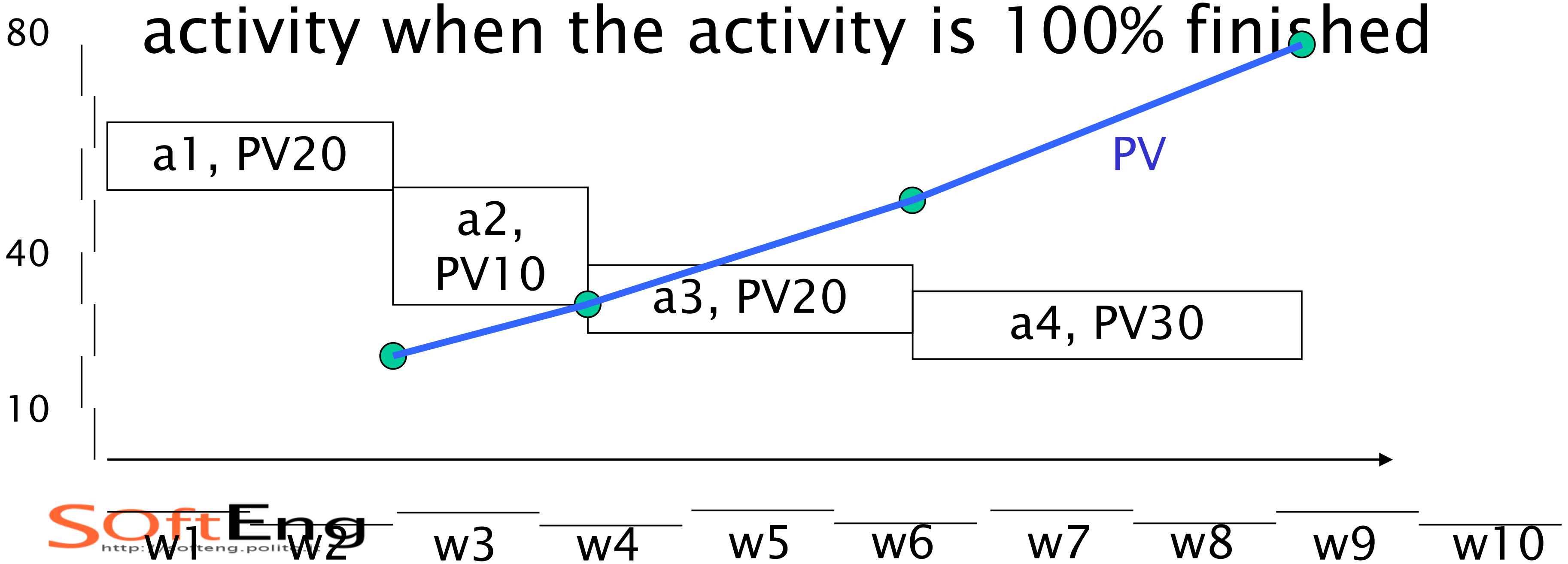
Earned value

- A technique to measure progress of a project
- Step 1: identify activities, assign a value to them (Planned Value, PV), schedule them



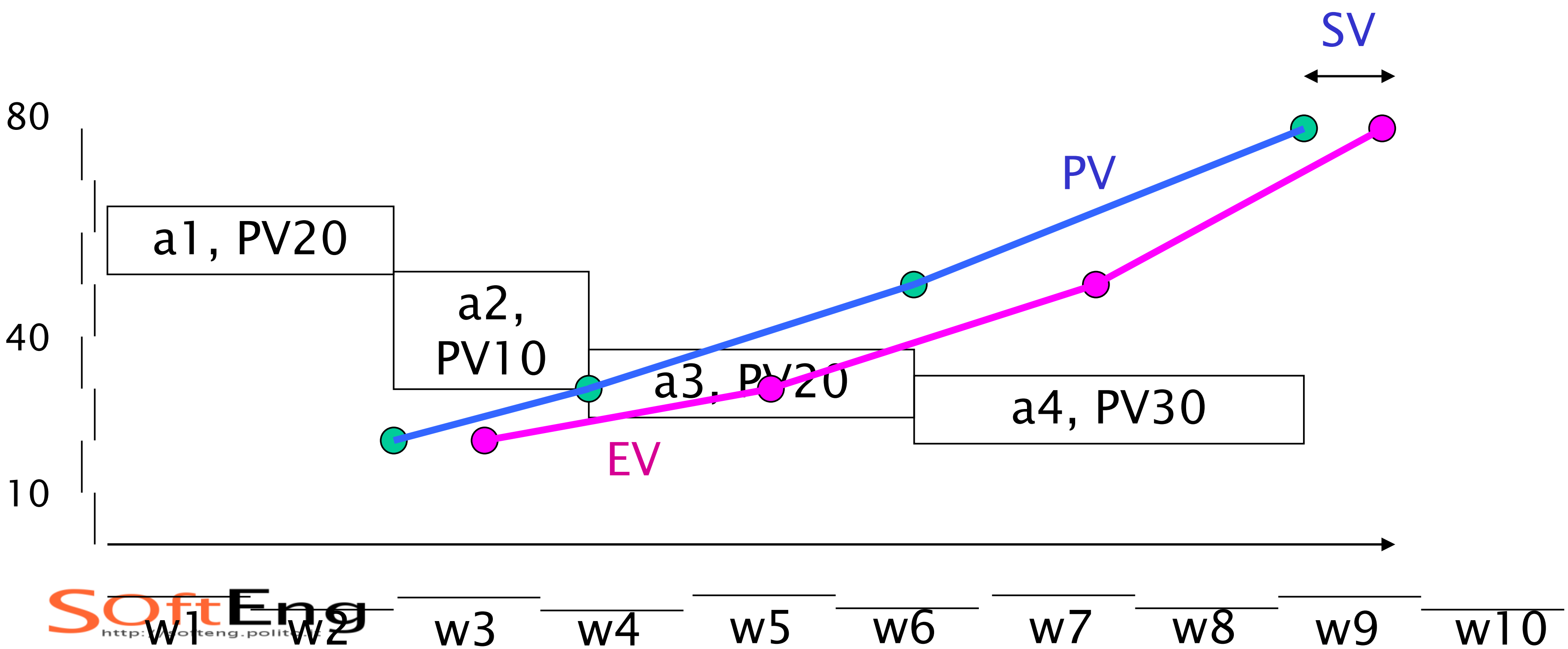
Earned value

- Step 2: define a rule to pass from PV to EV (rule1 0/100 or rule2 0/50/100)
  - With rule1, the project earns the PV of an activity when the activity is 100% finished



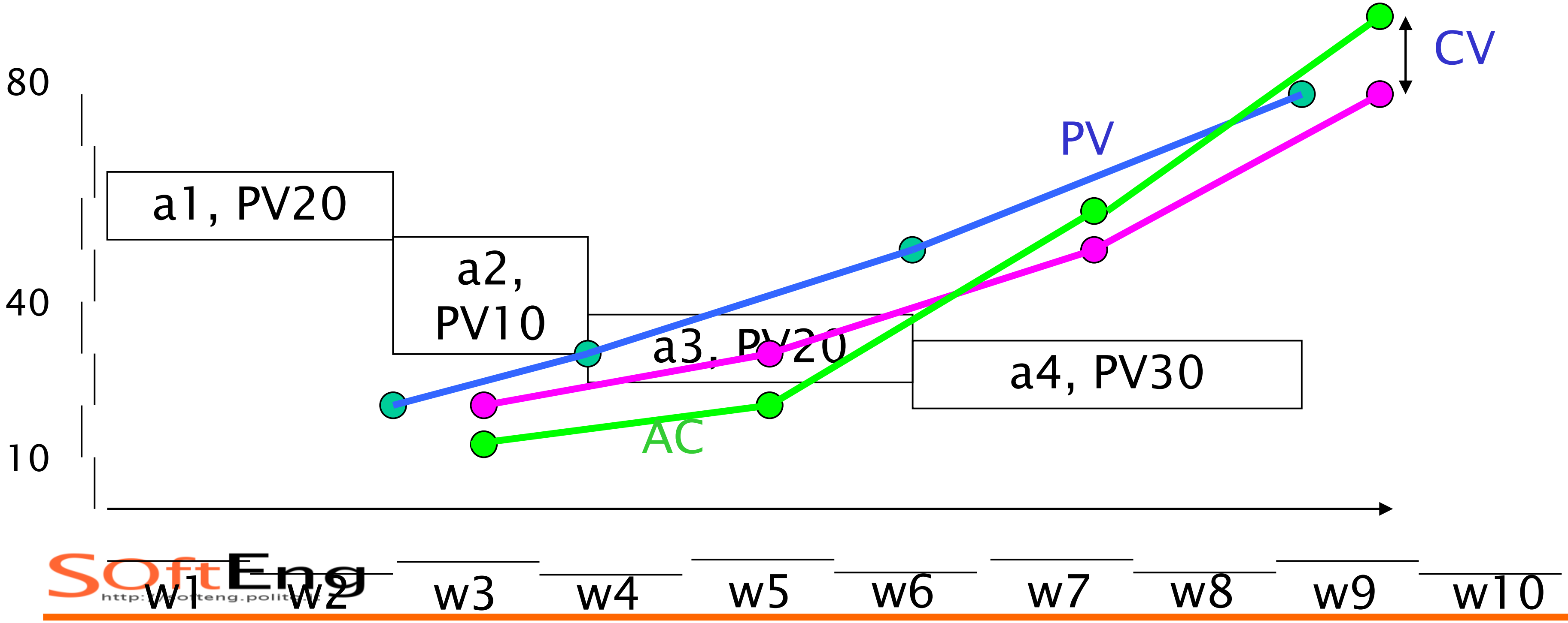
Earned value

- Step 3: start the project, measure EV and compare with PV

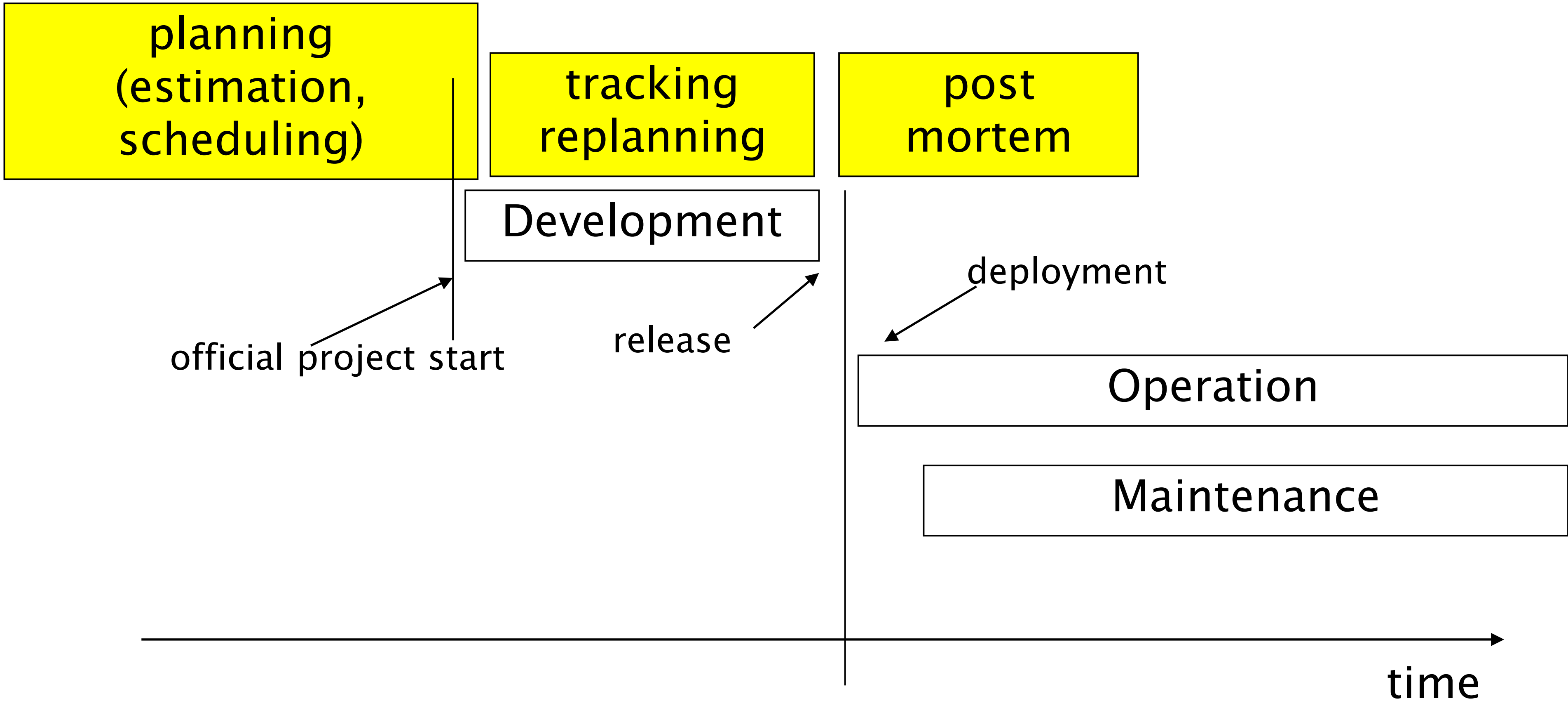


Earned value

- Step 4: compute also AC, actual cost



The PM activities



Post Mortem

Post mortem

- A form of organizational learning
- Collect key information from the project
  - Effort, faults – estimated and actual
  - Achievements
  - Problems and causes
- To make it available to other projects

PMA – learn from experience

- PMA (when used appropriately) PMA ensures that team members recognise and remember what they learned during a project.
- PMA identifies improvement opportunities and provides means to initiate sustained change.
- PMA provides qualitative feedback
- Two types
  - General PMA
  - Focused PMA – understanding and improving a project`s specific activity

PMA process

- Preparation
  - Study the project history to understand what has happened
  - Review all available documents
  - Determine goal for PMA
  - Example of goal: Identify major project achievements and further improvement opportunities.

PMA process cont.

- Data collection
  - Gather relevant project experience
  - Focus on positive and negative aspects
  - Semistructured interviews – pre-prepared list of questions
  - Facilitated group discussion
  - KJ sessions
    - Write down up to four positive and negative project experience on post-it notes.
    - Put the notes on a whiteboard
    - Re-arrange notes into groups and discuss them

PMA process cont.

- Analysis
  - Feedback session
    - Have we (analyser) understood what you (project member) told us, and do we have all the relevant facts?
  - Ishikawa diagram in a collaborative process to find the causes for positive and negative experiences
    - Draw an arrow on a whiteboard – which is label with experience
    - Add arrows with causes (the diagram will look like a fishbone)

PMA – results and experience

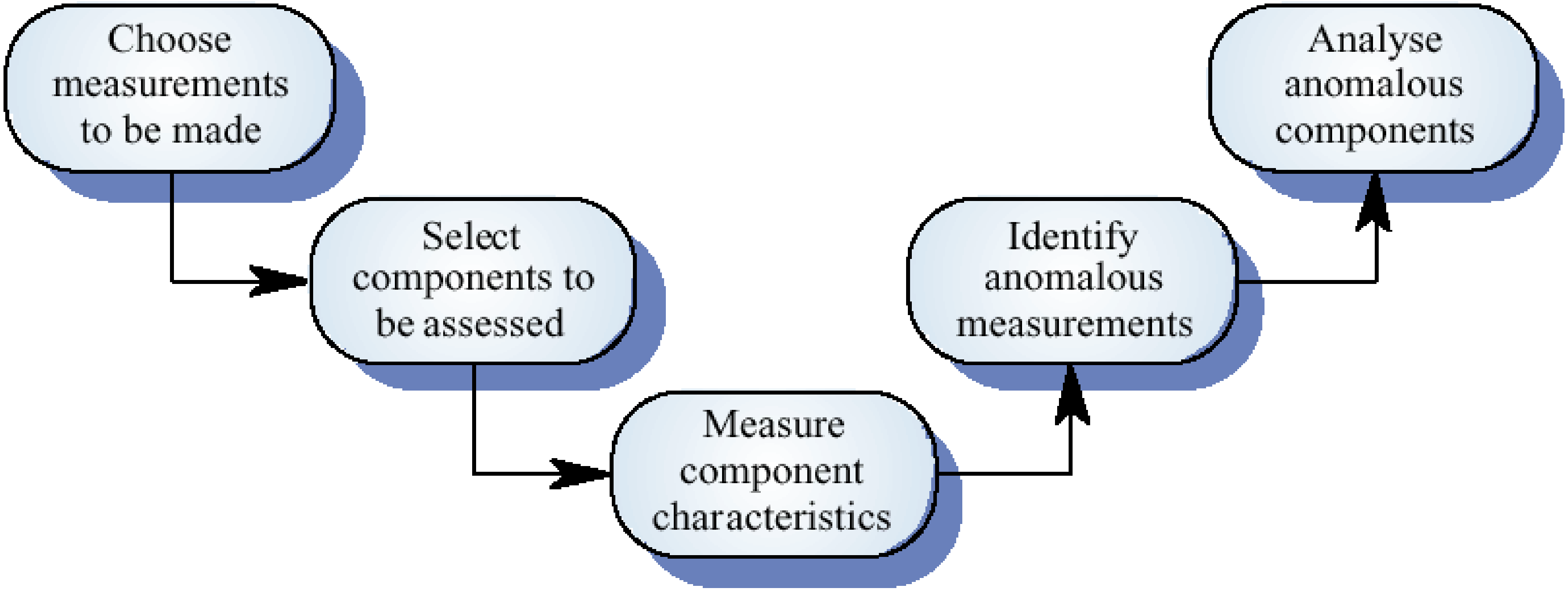
- Document the PMA results in a project experience report
  - Project description
  - Projects main problems, with description and Ishikawa diagrams
  - Project main success, with descriptions and Ishikawa diagrams
  - PMA meeting as an appendix (to let the reader see how the team discussed problems and successes)

Collecting and using measures

The measurement process

- A process should be defined and implemented to collect data, derive and analyze measures
- Data collected during this process should be maintained as an organisational resource
- Once a measurement database has been established, comparisons across projects become possible

Product measurement process



GQM

- Focus on few, important measures (top down)
- Never “collect everything, analyze later” (bottom up)
  - Too much data
  - Not meaningful

Goal – (similar to KPI)

- G1Satisfying customer
  - What is satisfaction?
    - Interviews
  - What is quality of product?
    - Defects after delivery

- G2 produce low cost product
  - What is cost
    - Cost of development

Typical indicators

- Effort (Cost)
- Size
- Defects after delivery
- Defects during development

GQM example

- Overall research question
  - Are UML Object diagrams useful?

Goal

- Object of study
  - UML Static structure diagrams
- Purpose
  - Evaluate
- Focus
  - Usefulness
- Point of view
  - Maintainer comprehending software
- Context
  - Master degree class

Data collection

- A metrics programme should be based on a set of product and process data
- Data should be collected immediately (not in retrospect) and, if possible, automatically
- Data should be controlled and validated as soon as possible

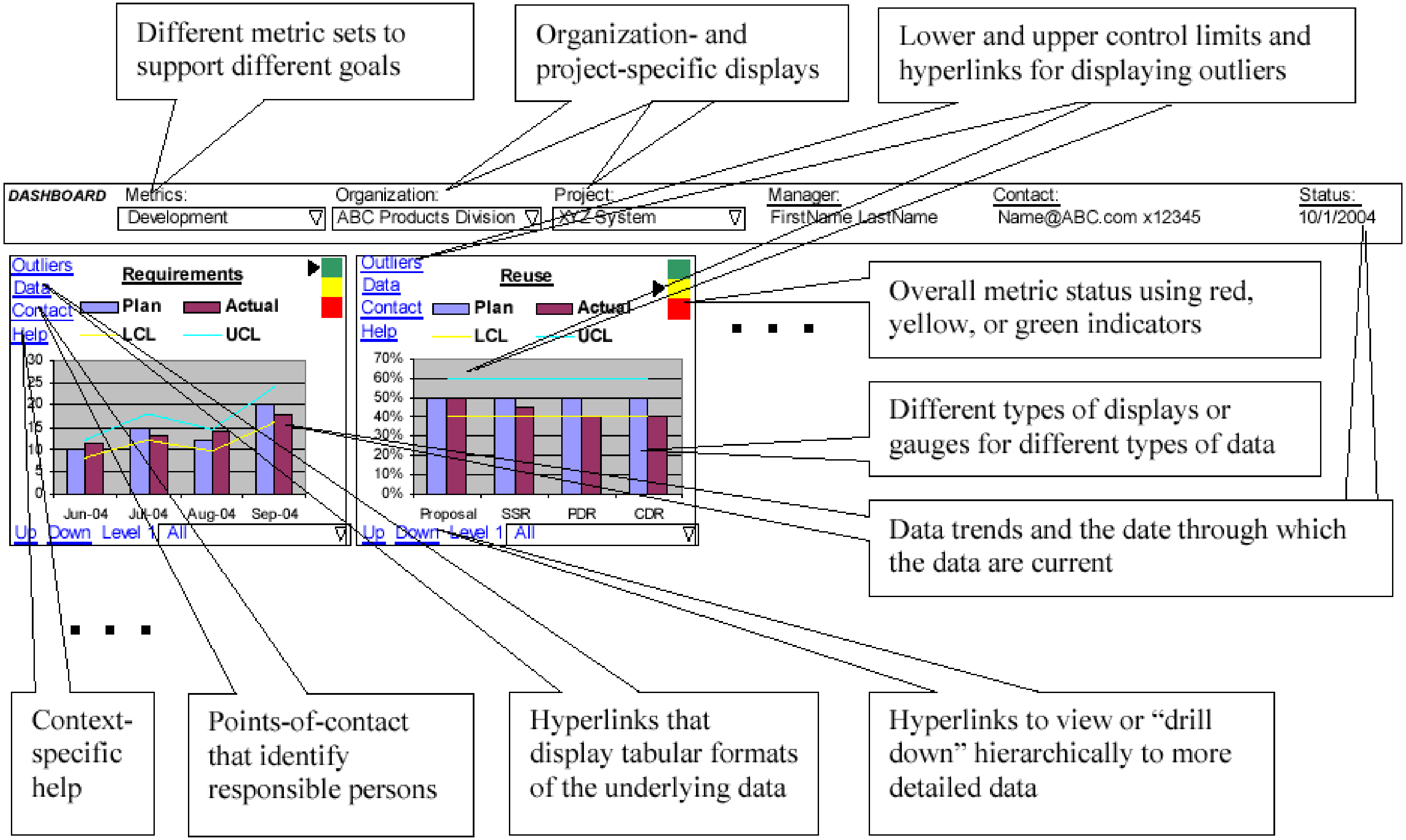
Data accuracy

- Don't collect unnecessary data
  - The questions to be answered should be decided in advance and the required data identified
- Tell people why the data is being collected
  - It should not be part of personnel evaluation
- Don't rely on memory
  - Collect data when it is generated not after a project has finished

Data presentation

- Reports
- Web reports
- Dashboard

Dashboard



Different metric sets to support different goals
Organization- and project-specific displays
Lower and upper control limits and hyperlinks for displaying outliers
DASHBOARD
Metrics:
Development
Organization:
ABC Products Division
Project:
XYZ System
Manager:
FirstName LastName
Contact:
Name@ABC.com x12345
Status:
10/1/2004
Outliers
Data
Contact
Help
Requirements
Plan
Actual
LCL
UCL
30
25
20
15
10
5
0
Jun-04
Jul-04
Aug-04
Sep-04
Up
Down
Level 1
All
Outliers
Data
Contact
Help
Reuse
Plan
Actual
LCL
UCL
70%
60%
50%
40%
30%
20%
10%
0%
Proposal
SSR
PDR
CDR
Up
Down
Level 1
All
Overall metric status using red, yellow, or green indicators
Different types of displays or gauges for different types of data
Data trends and the date through which the data are current
Context-specific help
Points-of-contact that identify responsible persons
Hyperlinks that display tabular formats of the underlying data
Hyperlinks to view or “drill down” hierarchically to more detailed data

Personnel

Project personnel

- Key activities requiring personnel:
  - requirements analysis
  - system design
  - software design
  - implementation
  - testing
  - training
  - maintenance
  - quality assurance

Choosing personnel

- ability to perform work
- interest in work
- experience with
  - similar applications
  - similar tools or languages
  - similar techniques
  - similar development environments
- training
- ability to communicate with others
- ability to share responsibility
- management skills

Work styles

- Extroverts: tell their thoughts
- Introverts: ask for suggestions
- Intuitives: base decisions on feelings
- Rationals: base decisions on facts, options

Organizational structure

- Depends on
  - backgrounds and work styles of team members
  - number of people on team
    - n people, max interactions = $n^2/2$
  - management styles of customers and developers
- Examples:
  - Chief programmer team
  - Egoless approach

Organizational structures

Highly structured

- high certainty
- repetition
- large project

Loosely structured

- uncertainty
- new technology
- small projects

Risk management

- Nothing is easy as it looks
- Everything takes longer than you expect
- And if anything can go wrong, it will, at the worst possible moment

[Murphy’s law]

Risk management

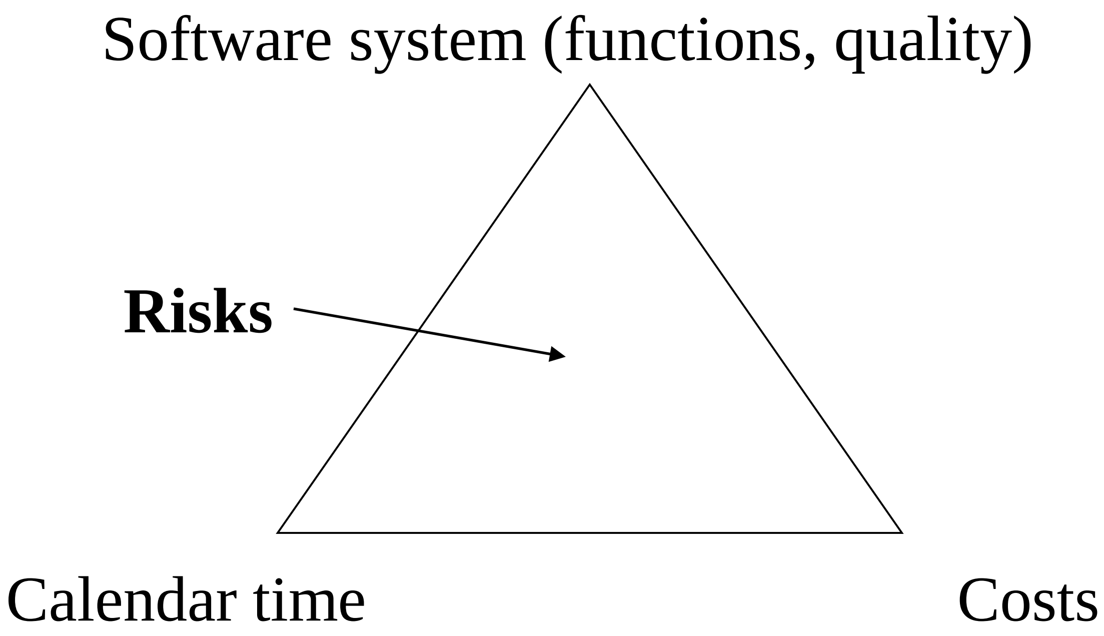
- Project Management for adults

If you don’t actively attack the risks,

they will actively attack you

Tom Gilb

Risk Management



Strategies

- Reactive
  - "Indiana Jones school of risk management"
  - Risk management = Crisis management ("fire-fighting mode")
- Proactive

Risk management (proactive)

- Identify risks
- analyze them
- quantify effects
- define strategies and plans to handle them

Risk

- Future event that can have (bad) impact on project

Risk categories

- Project
- Technical
- Business

- Known
- Predictable
- Unknown

Project Risks

- Regarding (ill defined) project plan
  - budget, personnel, timings, resources, customers
- Regarding management
  - No management support
  - Missing budget or people

Technical risks

- Regard feasibility of product
  - Design, interfaces, verification, ..

Business risks

- Regarding market or company
  - No market for the product (*market risk*)
  - Product not in scope with company plans (*strategic risk*)
  - Sales force does not know how to sell the product (*sales risk*)

Known risks

- Identified analyzing project characteristics
- Ex:
  - Unrealistic deadlines
  - No requirements
  - No focus
  - Poor development environment

Predictable risks

- Identified analyzing previous projects
- Ex.
  - Personnel turnover
  - Poor communication with customer

Unknown

- Hard or impossible to identify..

Boehm’s top ten risk items

- Personnel shortfalls
- Unrealistic schedules and budgets
- Developing the wrong functions
- Developing the wrong user interfaces
- Gold-plating
- Continuing stream of requirements changes
- Shortfalls in externally-performed tasks
- Shortfalls in externally-furnished components
- Real-time performance shortfalls
- Straining computer science capabilities

Other common risks

- instability of COTS (Commercial Off-The-Shelf) components/products
- interface with legacy
- stability of development platform (hw + sw)
- limitations of platform
- multi-site development
- use of new methodologies / technologies
- standards, laws
- development team involved in other activities
- communication/language problems

Risk management terms

- Risk impact: the loss associated with the event
- Risk probability: the likelihood that the event will occur
- Risk control: the degree to which we can change the outcome

Risk exposure = (risk probability) x (risk impact)

RM Process

- 1- Risk assessment
  - identification
  - analysis
  - ranking
- 2- Risk control
  - planning
  - monitoring

Identification

- identify risks
  - checklist, taxonomies, questionnaires
    - PMI (Project Management Institute, PMBOK)
    - SEI (SEI-93-TR-06)
      - ex: technical, management, business risks
  - brainstorming
  - experience

Analysis

- probability
  - very high, high, medium, low, very low
- impact
  - catastrophic, critical, marginal, negligible
- exposure
  - probability * impact

Exposure

Impact/ probability	Very high	High	Medium	Low	Very low
Catastrophic	High	High	Moderate	Moderate	Low
Critical	High	High	Moderate	Low	Null
Marginal	Moderate	Moderate	Low	Null	Null
Negligible	Moderate	Low	Low	Null	Null

Ranking

- By exposure
- by qualitative assessments
  - only higher exposure risks are handled

RM Process

- 1- Risk assessment
  - identification
  - analysis
  - ranking
- 2- Risk control
  - planning
  - monitoring

Planning

- For selected risks (high in exposure)
  - define corrective actions
  - evaluate cost, decide if acceptable
  - insert actions in project plan

Three strategies for risk reduction

- avoiding the risk: change requirements for performance or functionality
- transferring the risk: transfer to other system, or buy insurance
- assuming the risk: accept and control it

risk leverage = difference in risk exposure divided by cost of reducing the risk

Ex.

- ABS for car, software controlled. More flexible, but risk of failure from software
  - Avoiding. No software controlled
  - Transfer. Insurance.
  - Assuming. Develop software with best techniques, apply redundancy.

Ex.

- Risk leverage
  - ABS, software developed normally
    - cost 100KEuro,
    - risk exposure = 10^{-3} * 1M Euro
  - ABS, software developed best techniques
    - cost 1M Euro,
    - risk exposure = 10^{-6} * 1M Euro
  - Risk leverage

    (10^{-3} * 1M Euro – 10^{-6} * 1M Euro) / (1M – 100k)Euro

Company profiles and risk handling styles

- project owner – takes charge of risk
- fixed price contract
- work provider – no interest in risk

Monitoring

- follow project plan, including corrective actions
- monitor status of risks
- identify new risks, assess them, update ranking

Monitoring (2)

- Is part of PM that has to consider also
  - risk log (document)
  - risk reviews (activities)
    - also with external assessors
    - can be coupled with project reviews

Risk log

Risk	Probability	Impact	Exposure	Action	Status
hw platform not available	high	Critical	high	Add software Layer compatible with other platforms	Under control

Actions for risks

- Personnel shortfalls
  - hire the best, the most suitable, training, team building, technical reviews
- unrealistic schedules and budget
  - more detailed plans, iterative process, reuse, new plans
- instability of components (COTS)
  - qualification, detailed analysis of product and vendor, software layer.

- inadequate requirements
  - prototyping, JAD, iterative process, include user representative in process
    - Joint Application Development
- inadequate user interface
  - study user needs, usability analysis, prototyping
- requirement changes
  - suitable design, iterative process, rigid change control

- Interface with legacy
  - reengineering, encapsulation, incremental change
- subcontractors
  - contracts and payments, team building, assessments before and during
- new technologies
  - prototyping, cost benefit analysis

References

- www.pmi.org project management institute
- www.sei.cmu.edu
- www.ifpug.org
- http://www.itmpi-journal.com
- www.fhg.iese.de – Fraunhofer IESE
- Rapid Development – Taming Wild Software Schedules, Steve McConnell, Microsoft Press, 1996
- Software Engineering Risk Management, Dale Walter Karolak, IEEE Computer Society Press, 1996
- Assessment and Control of Software Risks , Caper Jones, Yourdon Press, 1994
- Software Risk Management – Principles and Practices, Barry W.Boehm, IEEE Software, Vol 8, No. 1, Jan 1991, PP32-41
- Taxonomy-Based Risk Identification, M.J.Carr et al., CMU/SEI-93-TR-06, SEI, 1993
- Www.riskwatch.com – Risk management tools