HY 351: Ανάλυση και Σχεδίαση Πληροφοριακών Συστημάτων
CS 351: Information Systems Analysis and Design

Project Management

Lecture : 5
Date : 11-10-2005

Yannis Tzitzikas
University of Crete, Fall 2005

Outline

- Introduction
  - Resource Allocation and Planning
  - What is Project Management
- Steps of Project Management
  - Identify Project Size
    - **the function point approach**
  - Create and Manage the WorkPlan
    - **Critical Path Analysis (CPA)**
    - **Gantt charts**
  - Staffing the Project
  - Coordinating project activities
- Vars
  - Managing iterations
  - Timeboxing

Resource Allocation and Planning

Systems development projects are similar to any other project in their need for sound management to ensure that they are completed within budget and on time.

- Large development projects
  - involve many different persons, some with specialized skills and
  - comprise activities whose sequence is important
- We need tools and techniques to support the process of project management, i.e. to allow
  - The **estimation** of money, time and people required
  - Assisting the **revision** of these estimates as a project continues
  - Helping to **track** and **manage** the tasks and activities carried out by a team of software developers

What is Project Management?

- **Project Management**: the process of planning and controlling the development of a system within a specified time frame at a minimum cost with the right functionality
- Key person: **Project manager**

The Steps of Project Management

[A] Identifying Project Size
[B] Creating and Managing the WorkPlan
[C] Staffing the Project
[D] Coordinating project activities

The Steps of Project Management

[A] Identifying Project Size
[B] Creating and Managing the WorkPlan
[C] Staffing the Project
[D] Coordinating project activities

[A] Identifying (estimating) Project Size

- This is usually based on **experience**
  - because we haven't yet analysed the requirements
- Rule of thumb:
  - use the amount of time spent in the planning phase to predict the time required for the entire project
  - ***commonly the planning phase takes 15% of the total time***
- However it could also involve the measurement of various aspects of the system using software metrics

(*Software Metrics*)

Software metric:

- A measure of some aspect of software development, either at project level (cost, duration), or at the level of application (size, complexity)

Types of Software Metrics:

- One distinction (past vs future)
  - Result (or control) metrics
  - Predictor metrics
- Another distinction
  - Process metrics
  - Product metrics

[A] Identifying (estimating) Project Size
The Function Point Approach

Steps:

(a) Estimate system size
- (a1) estimate function points
- (a2) estimate lines of code

(b) Estimate effort required
- in man months

(c) Estimate time required
- in months

Function Point Approach:
(a1) Estimate Function Points

Count the number of
- inputs (e.g. data entry screens)
- outputs (e.g. reports)
- queries (e.g. db queries)
- program interfaces

Example
- ***6 inputs***
- ***19 outputs***
- ***10 queries***
- ***15 files***
- ***3 program interfaces***

Remark:
- At the beginning we don't know exactly how many they are. So we make an estimation. In the next phases of the project we can revise them. Don't forget that planning is a continuous process.

Function Point Approach:
(a1) Partition them according to their complexity

Then we partition them according to their complexity to: low / medium / high

Remark:
- don't take into account here the skills of the team.

Example
- ***6 inputs*** ***(3 low, 2 medium, 1 high)***
- ***19 outputs*** ***(4 low, 10 medium, 5 high)***
- ***10 queries*** ***(7 low, 0 medium, 3 high)***
- ***15 files*** ***(0 low, 15 medium, 0 high)***
- ***3 program interfaces*** ***(1 low, 0 medium, 2 high)***

(a1.1) Estimate Function Points
Quantify the complexity

Multiply them with a complexity index

	Complexity			
Description	Low	Medium	High	Total
Inputs	__x 3	__x 4	__x 6	____
Outputs	***4*** x 4	***10*** x 5	***5*** x 7	**101**
Queries	__x 3	__x 4	__x 6	____
Files	__x 7	__x 10	__x 15	____
Program Interfaces	__x 5	__x 7	__x 10	____
TOTAL UNADJUSTED FUNCTION POINTS (**TUFP**)				***338***

Each team may have low or high expertise on some things.

How we could take into account the expertise of the team ?

Adjusting the Processing Complexity

TAFP = TUFP * APC

- TAFP: Total Adjusted Function Points
- TUFP: Total Unadjusted Function Points
- APC: Adjusted Processing Complexity

A simple rule is to assume that APC is equal to

- 0.65 for very simple (for the team) systems
- 1 for normal systems
- 1.35 for complex systems

A more refined method to express the expertise of the team

Each team may have low or high expertise on some things.
We could capture this with a table of the following form:

	Scale of 1 to 3
Data Communications	__0__
Heavy Use Configuration	__0__
Transaction Rate	__0___
End-User efficiency	__0___
Complex Processing	__0___
Installation Ease	__2___
Multiple sites	__0___
Performance	__0___
Distributed functions	__3__
On-line data entry	__0___
On-line update	__0___
Reusability	__0___
Operational Ease	__0___
Extensibility	__0___
Project Complexity (PC)	___5__

0: no effect on processing complexity
…
3: great effect on processing complexity

This characterizes the expertise of a team

APC = 0.65 + (0.01 * PC)

(a2) Estimate lines of code: Converting Function Points to Lines of Code

Language	LOC/Function Code Point
C	130
COBOL	110
JAVA	55
C++	50
Turbo Pascal	50
Visual Basic	30
PowerBuilder	15
HTML	15
Packages (e.g., Access, Excel)	10-40

Source: Capers Jones, Software Productivity Research

(b). Estimate **effort** required (in person months)

- This depends on the system's size and the production rates of the team
- One of the popular algorithms to convert a lines-of-code estimate to a person-month estimate is the COCOMO model (W. Boehm).
- For small to moderate-size business software projects (I.e. 100.000 lines of code and 10 or fewer programmers)

effort (in person months) = 1.4 * thousands of lines of code

– so, 20 working days = 1.400 lines => 1 day = 70 lines

(c) Estimate **Time Period** (in months)

- Historical data or estimation software can be used as aids for this.
- A rule of thumb

  schedule time (months) = 3 * person-months ^(1/3)

  – E.g. a project with effort 14 months should be scheduled to take a little more than 7 months to complete

  – // note: Adhamls law on parallelism related

The Steps of Project Management

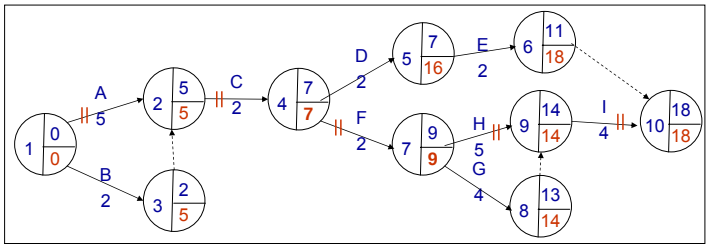
[A] Identifying Project Size
[B] Creating and Managing the WorkPlan
[C] Staffing the Project
[D] Coordinating project activities

[B]. Creating and Managing the WorkPlan

- Tools and techniques
  - **Critical Path Analysis (CPA)**
  - **Gantt charts**

Critical Path Analysis (CPA)

- Originally known as PERT (Project Evaluation and Review Technique)
  - Developed in the late 50s for use on major weapons development projects for the US Navy



The steps for preparing a CPA chart

1. List all project activities and milestones
2. Determine the dependencies among the activities
3. Estimate the duration of each activity
4. Draw the CPA chart

1. List all project activities and milestones

How to identify tasks:

- Top-down approach
  - Identify highest level tasks
  - Break them into increasingly smaller units
    - e.g. Dennis page 94 (steps to organize an IT course)
    - this is the **work breakdown structure** (WBS)
- Methodology
  - Using standard list of tasks (e.g. from the adopted methodology)

Example

1. List all project activities and milestones

Activity	Description	Milestone
A	Interview users	2
B	Prepare use cases	3
C	Review use cases	4
D	Draft screen layouts	5
E	Review screens	6
F	Identify classes	7
G	CRC Analysis	8
H	Prepare draft class diagram	9
I	Review class diagram	10

Example

1. List all project activities and milestones
2. Determine the dependencies among the activities

Activity	Description	Milestone	Preceding activities
A	Interview users	2	-
B	Prepare use cases	3	-
C	Review use cases	4	A, B
D	Draft screen layouts	5	C
E	Review screens	6	D
F	Identify classes	7	C
G	CRC Analysis	8	F
H	Prepare draft class diagram	9	F
I	Review class diagram	10	G, H

Example

1. List all project activities and milestones
2. Determine the dependencies among the activities
3. Estimate the duration of each activity

Expected duration of a task:

ED = (MOT + MPT + 4*MLT)/6

- MOT: most optimistic time for the completion of an activity
  - assumption: No delay will occur
- MLT: most likely time for the completion of an activity
  - Assumption: all things will go wrong
- MPT: most pessimistic time for the completion of an activity

Example

1. List all project activities and milestones
2. Determine the dependencies among the activities
3. Estimate the duration of each activity

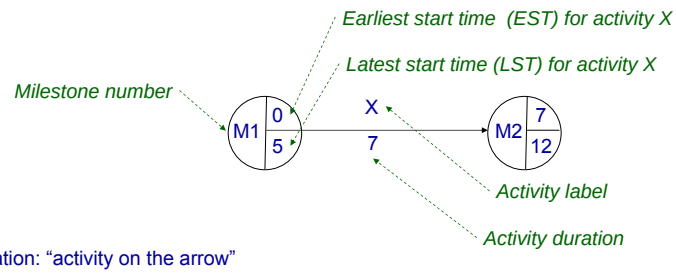
Activity	Description	Milestone	Preceding activities	Expected duration
A	Interview users	2	-	5
B	Prepare use cases	3	-	2
C	Review use cases	4	A, B	2
D	Draft screen layouts	5	C	2
E	Review screens	6	D	2
F	Identify classes	7	C	2
G	CRC Analysis	8	F	4
H	Prepare draft class diagram	9	F	5
I	Review class diagram	10	G, H	4

A Workplan Example

Work Plan Information	Example
Name of task	Perform economic feasibility
Start date	Oct 11, 2005
Completion date	Oct 20, 2005
Person assigned	Student X
Deliverable(s)	Cost-benefit analysis
Completion status	Open
Priority	High
Resources needed	Spreadsheet
Estimated time	4 hours
Actual time	2 hours

CPA chart

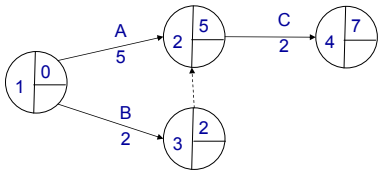
- 1. List all project activities and milestones
- 2. Determine the dependencies among the activities
- 3. Estimate the duration of each activity
- 4. Draw the CPA chart



// notation: "activity on the arrow"

Example: : CPA chart with earliest times

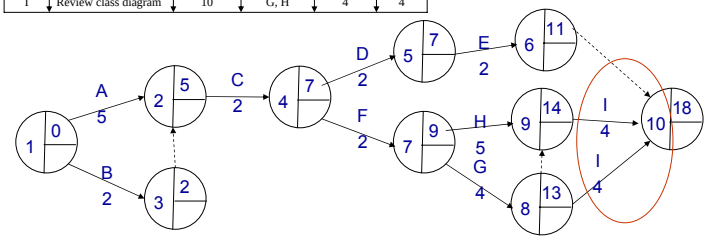
Acti-vity	Description	Milestone	Preceding activities	Expected duration	Staffing
A	Interview users	2	-	5	2
B	Prepare use cases	3	-	2	See A
C	Review use cases	4	A, B	2	3
D	Draft screen layouts	5	C	2	2
E	Review screens	6	D	2	2
F	Identify classes	7	C	2	3
G	CRC Analysis	8	F	4	3
H	Prepare draft cl.Diagr.	9	F	5	3
I	Review class diagram	10	G, H	4	4



Example: CPA chart with earliest times

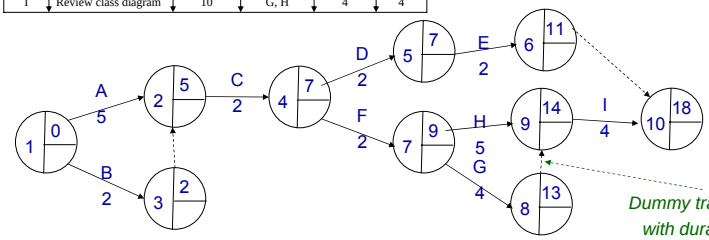
Acti-vity	Description	Milestone	Preceding activities	Expected duration	Staffing
A	Interview users	2	-	5	2
B	Prepare use cases	3	-	2	See A
C	Review use cases	4	A, B	2	3
D	Draft screen layouts	5	C	2	2
E	Review screens	6	D	2	2
F	Identify classes	7	C	2	3
G	CRC Analysis	8	F	4	3
H	Prepare draft cl.Diagr.	9	F	5	3
I	Review class diagram	10	G, H	4	4

forward pass



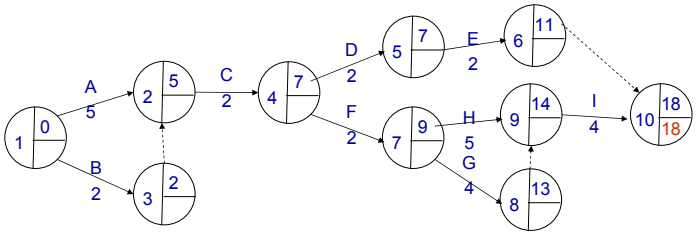
Example: CPA chart with earliest times

Acti-vity	Description	Milestone	Preceding activities	Expected duration	Staffing
A	Interview users	2	-	5	2
B	Prepare use cases	3	-	2	See A
C	Review use cases	4	A, B	2	3
D	Draft screen layouts	5	C	2	2
E	Review screens	6	D	2	2
F	Identify classes	7	C	2	3
G	CRC Analysis	8	F	4	3
H	Prepare draft cl.Diagr.	9	F	5	3
I	Review class diagram	10	G, H	4	4



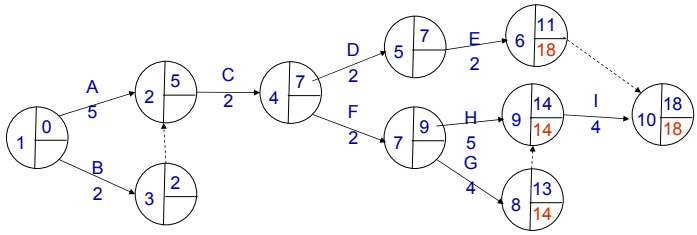
Example: adding latest times

backwards pass



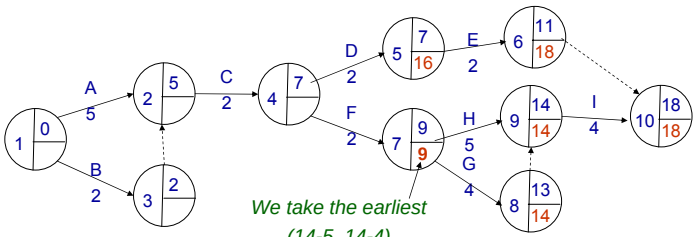
Example: adding latest times

backwards pass



Example: adding latest times

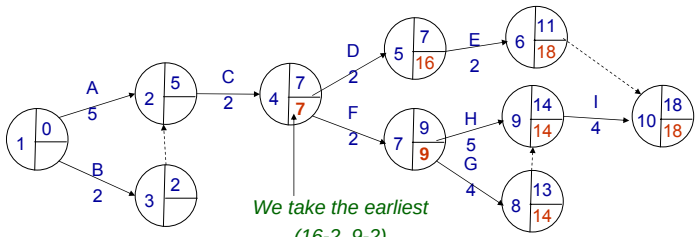
backwards pass



We take the earliest
(14-5, 14-4)

Example: adding latest times

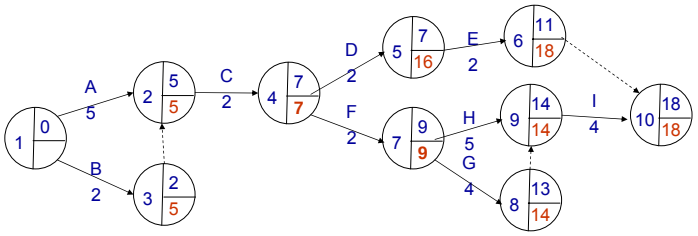
backwards pass



We take the earliest
(16-2, 9-2)

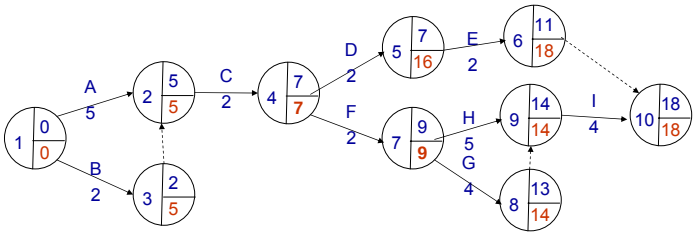
Example: adding latest times

backwards pass



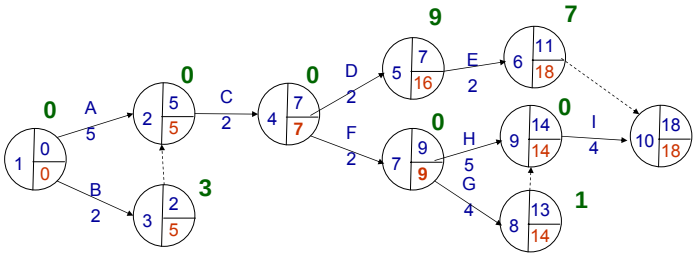
Example: adding latest times

backwards pass



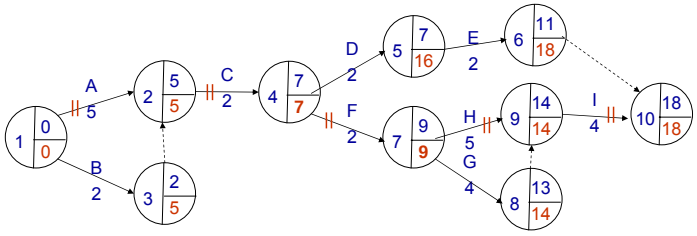
Slack (or float) **time** of an activity

- The difference between EST and LST
- It represents the time the time by which an activity can be delayed without affecting the overall duration of the project



CPA: Identify **critical path**

Critical path:
The path through all milestones that have a slack time of 0.



CPA summary

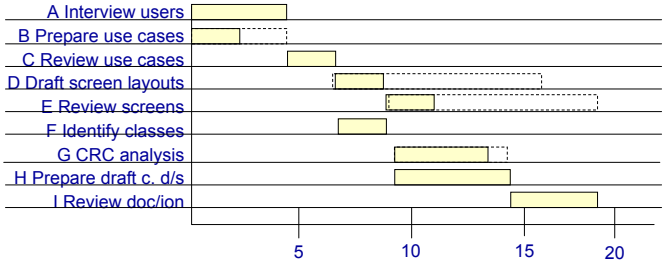
- CPA chart is an effective tool for identifying those activities whose completion is critical for the completion of a project on time
  - A delay on an activity on a critical path results in delay of the whose project
- Drawback of CPA charts
  - They do not show the overlap between activities (Gantt charts are better on this).

Gantt charts

- Horizontal axis: time
- Vertical axis: activities
- For each activity we draw a bar
  - length of bar: *ED* of the activity
  - Dash lines/boxes can be used to show its *slack time*

Example

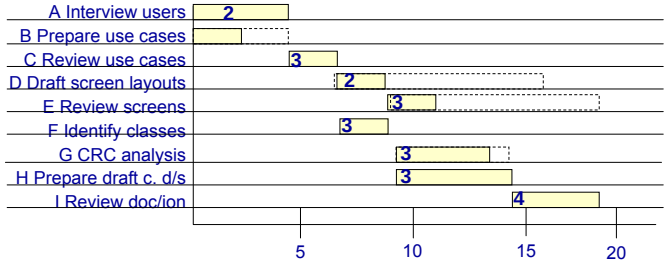
Acti-vity	Description	Milestone	Preceding activities	Expected duration	Staffing
A	Interview users	2	-	5	2
B	Prepare use cases	3	-	2	See A
C	Review use cases	4	A, B	2	3
D	Draft screen layouts	5	C	2	2
E	Review screens	6	D	2	2
F	Identify classes	7	C	2	3
G	CRC Analysis	8	F	4	3
H	Prepare draft cl.Diagr.	9	F	5	3
I	Review class diagram	10	G, H	4	4



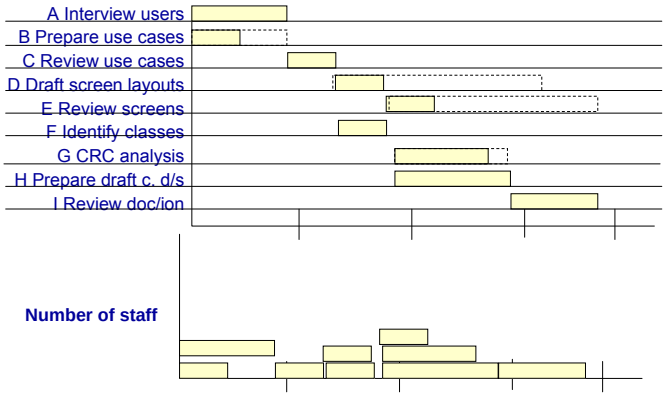
Example

We can annotate bars with mms (man months)

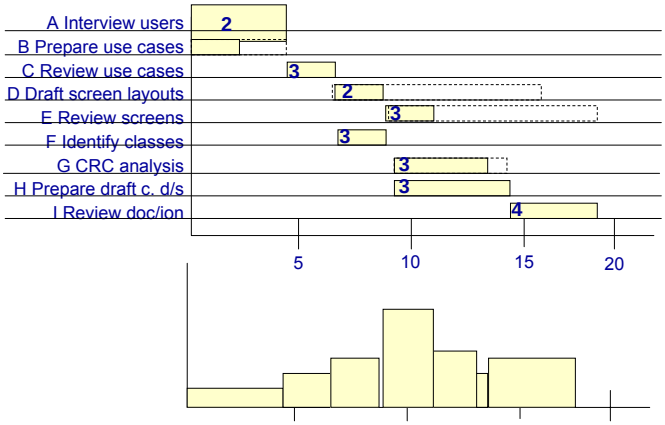
Acti-vity	Description	Milestone	Preceding activities	Expected duration	Staffing
A	Interview users	2	-	5	2
B	Prepare use cases	3	-	2	See A
C	Review use cases	4	A, B	2	3
D	Draft screen layouts	5	C	2	2
E	Review screens	6	D	2	2
F	Identify classes	7	C	2	3
G	CRC Analysis	8	F	4	3
H	Prepare draft cl.Diagr.	9	F	5	3
I	Review class diagram	10	G, H	4	4



From a Gantt chart to stacked bar graph



From a Gantt chart to stacked bar graph



The Steps of Project Management

[A] Identifying Project Size
[B] Creating and Managing the WorkPlan
[C] Staffing the Project
[D] Coordinating project activities

[C]. Staffing the Project

- ***how many persons should be assigned to the project?***
- ***How to assign project roles to team members?***

- Match people's skills with the needs of the project
- Develop a reporting structure for the team

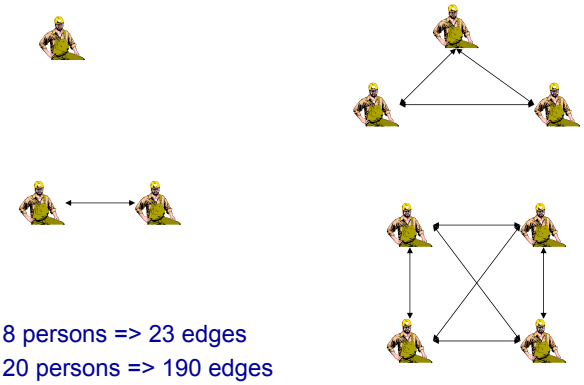
- may also include
  - motivating the team to meet the project's objectives
  - minimizing conflict among team members
    - Conflicts can be minimized by defining clearly the roles on a project.

Staffing the Project and Gantt charts

Resource smoothing

- Play with the slack times so to that to accommodate staff availability
  - E.g. reduce persons that should be hired

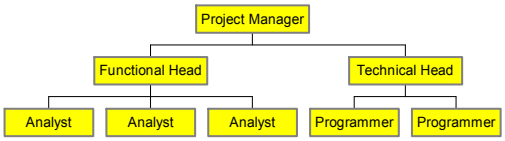
Increasing Complexity with Larger Teams



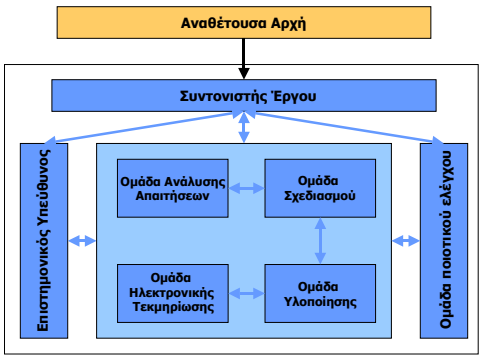
8 persons => 23 edges

20 persons => 190 edges

Staffing Plan (roles and reporting structure)



Ρόλοι και Αρμοδιότητες Ομάδας Έργου



The Steps of Project Management

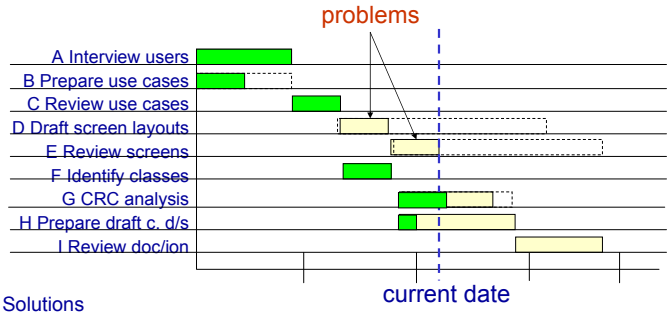
[A] Identifying Project Size

[B] Creating and Managing the WorkPlan

[C] Staffing the Project

[D] Coordinating project activities

Monitoring and Coordinating the project



Solutions

- Exploit slack times
- Allocate more staff (not always good, e.g. car repair)
- Delay the project
- Reduce the scope/quality of the project (after discussing with the client)

Behind schedule?

- When a critical path activity is behind schedule it may not be possible to regain the lost time
- In that case the project manager has only 2 options
  - **A: move forward the project deadline**
  - **B: reduce the scope/quality of the project**
- The choice should be discussed with the client
  - B: requires analysis in order to see what to omit in order to reach the deadline

Managing iteration

- Challenge: ***How to control the number of iterations?***
- Recall: prototyping is an incremental development activity.
- Prototyping activities should be accompanied by objectives that allows us to provide criteria to control the number of iterations
  - At the end of an iteration the prototype is evaluated against pre-defined objectives. In practice it may be difficult to determine whether the objectives of a prototyping activity have been met
    - e.g. UI with objective: make users happy (happiness is an endless process)
    - a tip: Continue the iterations until fewer than 5 cosmetic changes are requested on a single iteration
    - other tips:
      - The prototyping must be completed on December and must not exceed 30 developer-hours.

Dynamic Systems Development Methods (DSDM)

- Management and control framework for rapid application development (RAD)
  - The distinction between RAD and prototyping is blurred
    - RAD: build a working system rapidly
    - Prototyping: again build rapidly a system (but with partial functionality)
      - Typically to confirm some aspect of the requirements / arch/ etc. ..
- Key difference (wrt process control):
  - ***Instead of considering the requirements stable, DSDM fixes resources for the project, fixes the time available, and then sets out to deliver only what can be achieved within these contraints***

DSDM: Timeboxing

Key points:
- Fixed deadline
- Reduced functionality, if necessary
- Fewer "finishing touches"

- We need to **prioritize the requirements** that will be actioned during a timebox
- **MoSCoW rules** (**Must Should Could Want**) for requirements

Timeboxing

Timeboxing Steps

1/ Set delivery date
   Deadline should not be impossible
   Should be set by development group
2/ Prioritize features by importance
3/ Build the system core
4/ Postpone unfinished functionality
5/ Deliver the system with core functionality
6/ Repeat steps 3-5 to add refinements and enhancements

Classic Planning Mistakes

- Overly optimistic schedule
- Failing to monitor schedule
- Failing to update schedule
- Adding people to a late project

Tools for Project Management

- Microsoft Project
- Plan View
- PMOffice

- There are dozens of such tools
  - Take a look at *http://www.startwright.com/project1.htm*

Reading and References

- **Object-Oriented Systems Analysis and Design Using UML** (2nd edition) by S. Bennett, S. McRobb, R. Farmer, McGraw Hil, 2002. CHAPTER 21
- **Systems Analysis and Design with UML Version 2.0** (2nd edition) by A. Dennis, B. Haley Wixom, D. Tegarden, Wiley, 2005. CHAPTER 4
- **System Analysis and Design Methods** (6th edition) by Jeffrey L. Whitten, Lonnie D. Bentley and Kevin Dittman, McGraw-Hill, 2004, Chapter 4