BBM-382 – SOFTWARE ENGINEERING

SPRING 2021
Lecture 2

Assoc. Prof. Dr. Ayça KOLUKISA TARHAN
Dr. Tuğba ERDOĞAN

CHAPTER 22 – PROJECT MANAGEMENT

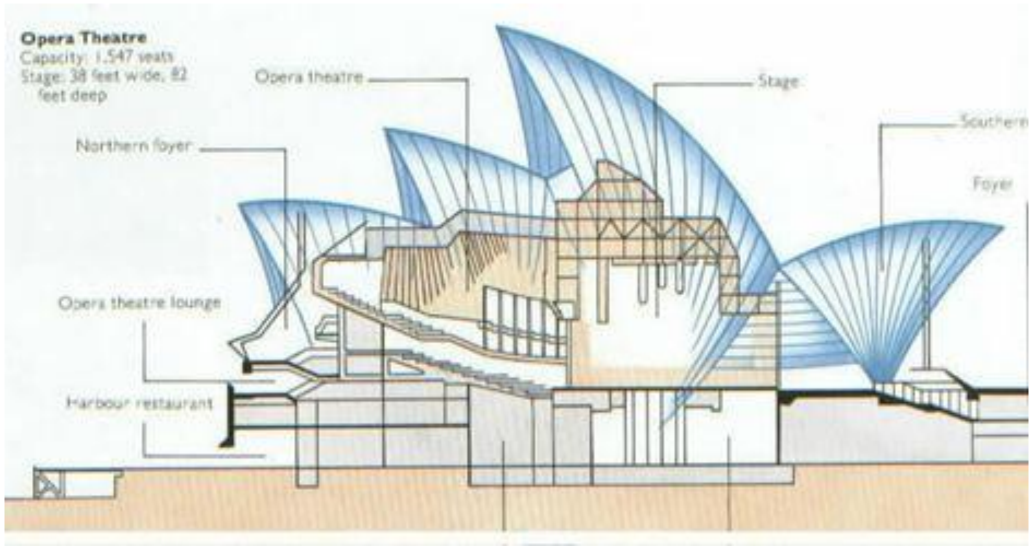
What is a Project?

- *"A project is a <u>temporary</u> endeavour undertaken to create a <u>unique</u> product or service"*

[Project Management Body of Knowledge-PMBOK, 2013]

- "Temporary"
  - Has a start and an end date
- "Unique"
  - Has specific attributes for its customers

Example: Civil Engineering



Example: Software Projects

- Developing MS Vista (a new operating system)
  - *Generic (commercial-off-the-shelf - COTS) software*
- Developing Airline Ticket Reservation System
  - *Custom software*

ANALYSIS
SPECIFICATION
IMPLEMENT
PROJECT
DESIGN
TESTING
DEVELOP

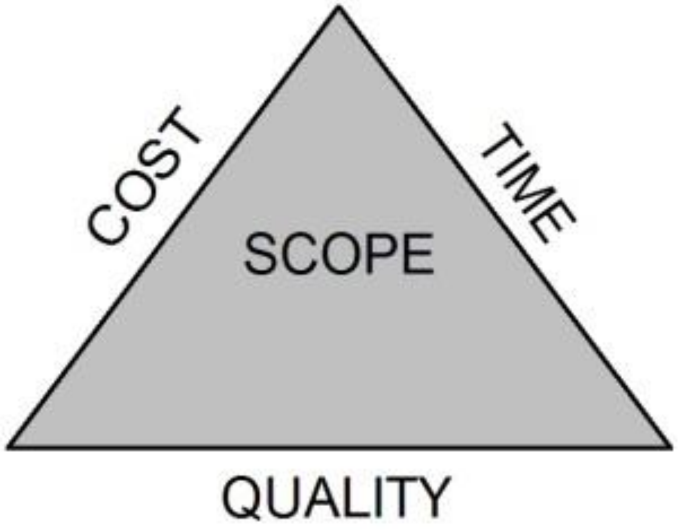
Product of the Sw.Eng.Lab. Project

- System to be developed: Social Interest eClub

What is Project Management?

- *"Project management is the application of knowledge, skills, tools, and techniques to project activities in order to meet project requirements"* [PMBOK, 2013]

- Requires alignment among:
  - The parameters of time, cost, quality, etc.
  - The needs of different stakeholders



Project Management Process

[PMBOK, 2013]

- *“Project Management is accomplished through the appropriate application of five process groups which are:*
  - *Initiating, Planning, Executing, Monitoring and Controlling, and Closing.”*

- *“Managing a project typically includes:*
  - *Identifying **requirements**,*
  - *Addressing the various needs, concerns, and expectations of the **stakeholders** as the project is planned and carried out,*
  - *Balancing the competing project **constraints** including (but not limited to):*
    - *Scope, Quality, Schedule, Budget, Resources, and Risk.”*

Project Management Process Groups
[PMBOK, 2013]

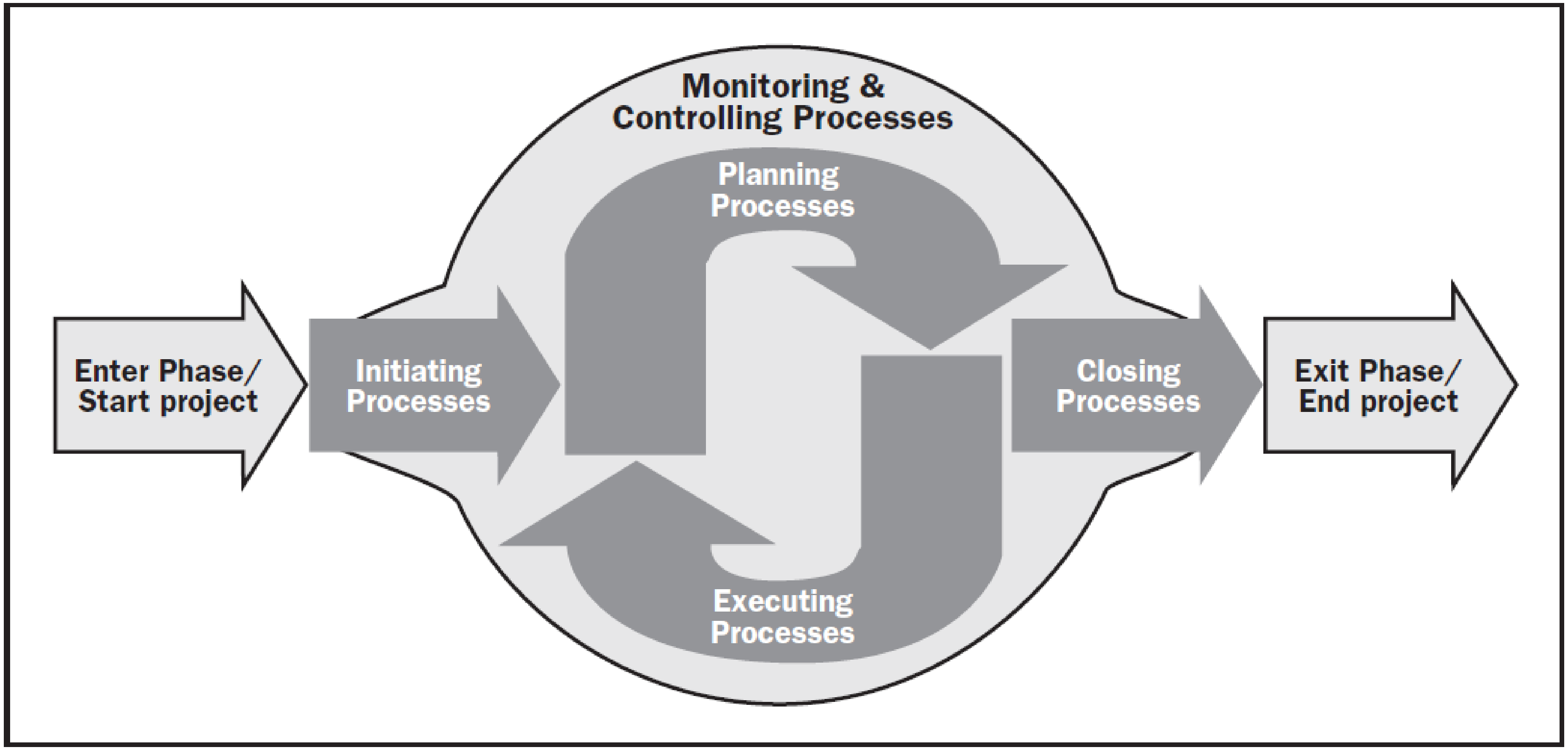


Figure 3-1. Project Management Process Groups

Project Management Processes with respect to time [PMBOK, 2013]

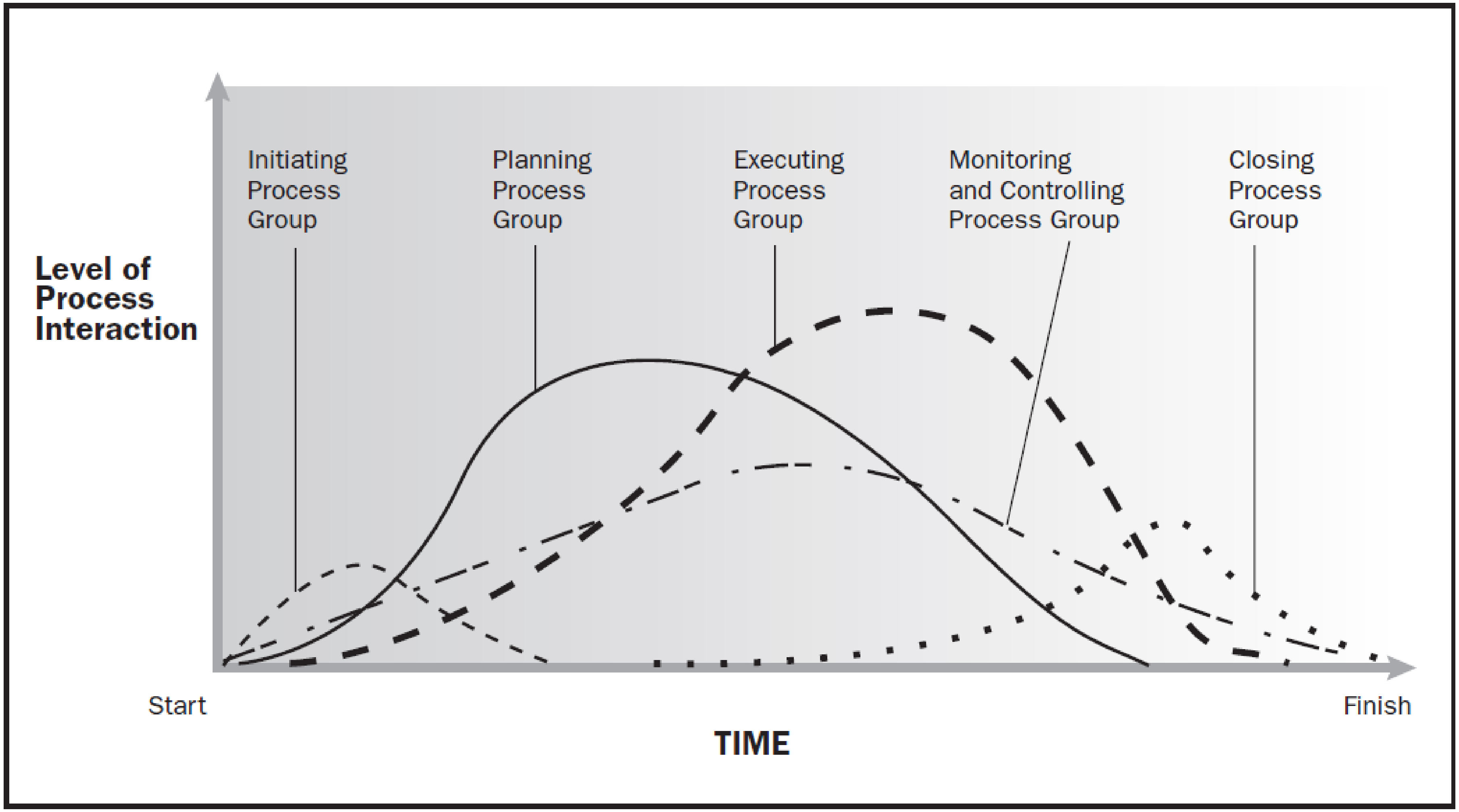


Figure 3-2. Process Groups Interact in a Phase or Project

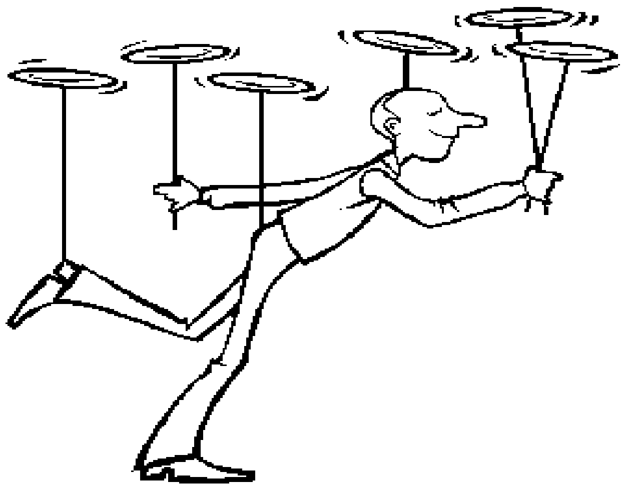
Software Project Management

- Concerned with activities involved in ensuring that software is delivered on time and on schedule and in accordance with the requirements of the organisations developing and procuring the software.

- Project management is needed because software development is always subject to budget and schedule constraints that are set by the organisation developing the software.

Success criteria

- Deliver the software to the customer at the agreed time.
- Keep overall costs within budget.
- Deliver software that meets the customer's expectations.
- Maintain a happy and well-functioning development team.



Software management distinctions

- The product is intangible.
  - Software cannot be seen or touched. Software project managers cannot see progress by simply looking at the artifact that is being constructed.
- Many software projects are 'one-off' projects.
  - Large software projects are usually different in some ways from previous projects. Even managers who have lots of previous experience may find it difficult to anticipate problems.
- Software processes are variable & organization specific.
  - We still cannot reliably predict when a particular software process is likely to lead to development problems.

We need management activities!

- Project planning
  - Project managers are responsible for planning. estimating and scheduling project development and assigning people to tasks.
- Reporting
  - Project managers are usually responsible for reporting on the progress of a project to customers and to the managers of the company developing the software.
- Risk management
  - Project managers assess the risks that may affect a project, monitor these risks and take action when problems arise.
- People management
  - Project managers have to choose people for their team and establish ways of working that leads to effective team performance
- Proposal writing
  - The first stage in a software project may involve writing a proposal to win a contract to carry out an item of work. The proposal describes the objectives of the project and how it will be carried out.

People management - Teamwork

- Most software engineering is a group activity.
  - The development schedule for most non-trivial software projects is such that they <u>cannot be completed by one person working alone</u>.
- A good group <u>is cohesive and has a team spirit</u>.
  - The people involved are motivated by the success of the group as well as by their own personal goals.
- <u>Group interaction</u> is a key determinant of group performance.
- Flexibility in group composition is limited.
  - Managers must do the best they can

  with available people.

People management - Group cohesiveness

- In a cohesive group, members consider the group to be more important than any individual in it.

- The advantages of a cohesive group are:
  - Group quality standards can be developed by the group members.
  - Team members learn from each other and get to know each other's work; Inhibitions caused by ignorance are reduced.
  - Knowledge is shared. Continuity can be maintained if a group member leaves.
  - Refactoring and continual improvement is encouraged. Group members work collectively to deliver high quality results and fix problems, irrespective of the individuals who originally created the design or program.

Risk management

- Risk management is concerned with <u>identifying risks</u> and drawing up plans to <u>minimise their effect</u> on a project.

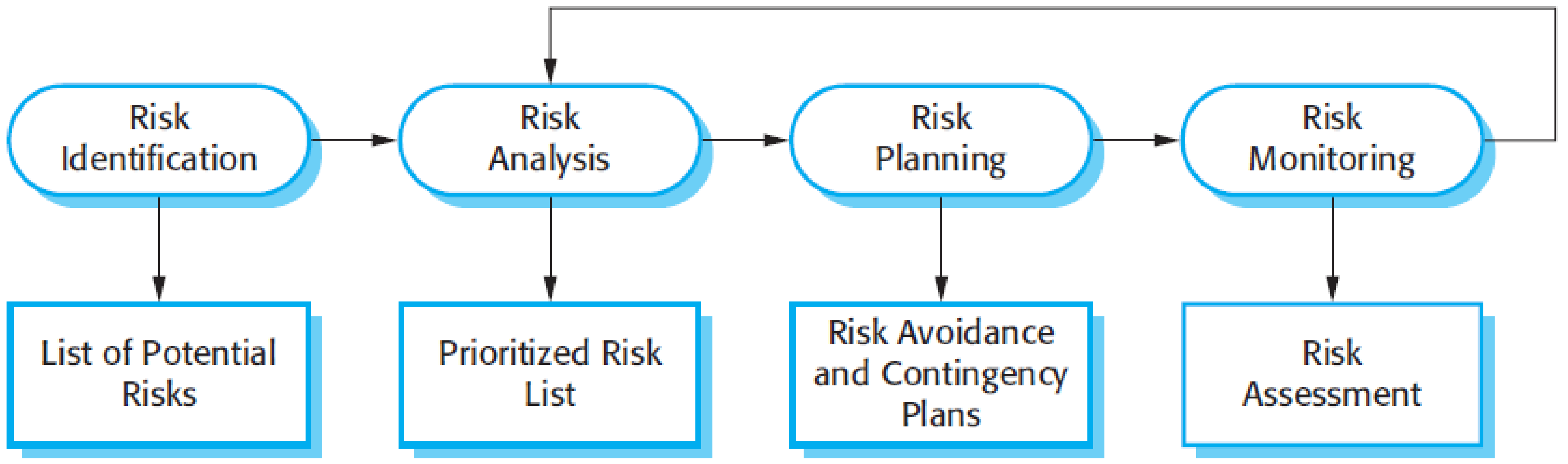
- A risk is a probability that some adverse circumstance will occur
  - Project risks affect schedule or resources;
  - Product risks affect the quality or performance of the software being developed;
  - Business risks affect the organisation developing or procuring the software.

Examples of common project, product, and business risks

Risk	Affects	Description
Staff turnover	Project	Experienced staff will leave the project before it is finished.
Management change	Project	There will be a change of organizational management with different priorities.
Hardware unavailability	Project	Hardware that is essential for the project will not be delivered on schedule.
Requirements change	Project and product	There will be a larger number of changes to the requirements than anticipated.
Specification delays	Project and product	Specifications of essential interfaces are not available on schedule.
Size underestimate	Project and product	The size of the system has been underestimated.
CASE tool underperformance	Product	CASE tools, which support the project, do not perform as anticipated.
Technology change	Business	The underlying technology on which the system is built is superseded by new technology.
Product competition	Business	A competitive product is marketed before the system is completed.

The risk management process

- Risk identification - Identify project, product and business risks;
- Risk analysis - Assess the likelihood and consequences of these risks;
- Risk planning - Draw up plans to avoid or minimise effects of the risk;
- Risk monitoring - Monitor the risks throughout the project.



Risk identification

- May be a team activities or based on the individual project manager’s experience.
- A checklist of common risks may be used to identify risks in a project
  - Technology risks
  - People risks
  - Organisational risks
  - Requirements risks
  - Estimation risks
  - Tool risks

Examples of different risk types

Risk type	Possible risks
Technology	The database used in the system cannot process as many transactions per second as expected. (1) Reusable software components contain defects that mean they cannot be reused as planned. (2)
People	It is impossible to recruit staff with the skills required. (3) Key staff are ill and unavailable at critical times. (4) Required training for staff is not available. (5)
Organizational	The organization is restructured so that different management are responsible for the project. (6) Organizational financial problems force reductions in the project budget. (7)
Tools	The code generated by software code generation tools is inefficient. (8) Software tools cannot work together in an integrated way. (9)
Requirements	Changes to requirements that require major design rework are proposed. (10) Customers fail to understand the impact of requirements changes. (11)
Estimation	The time required to develop the software is underestimated. (12) The rate of defect repair is underestimated. (13) The size of the software is underestimated. (14)

Risk analysis

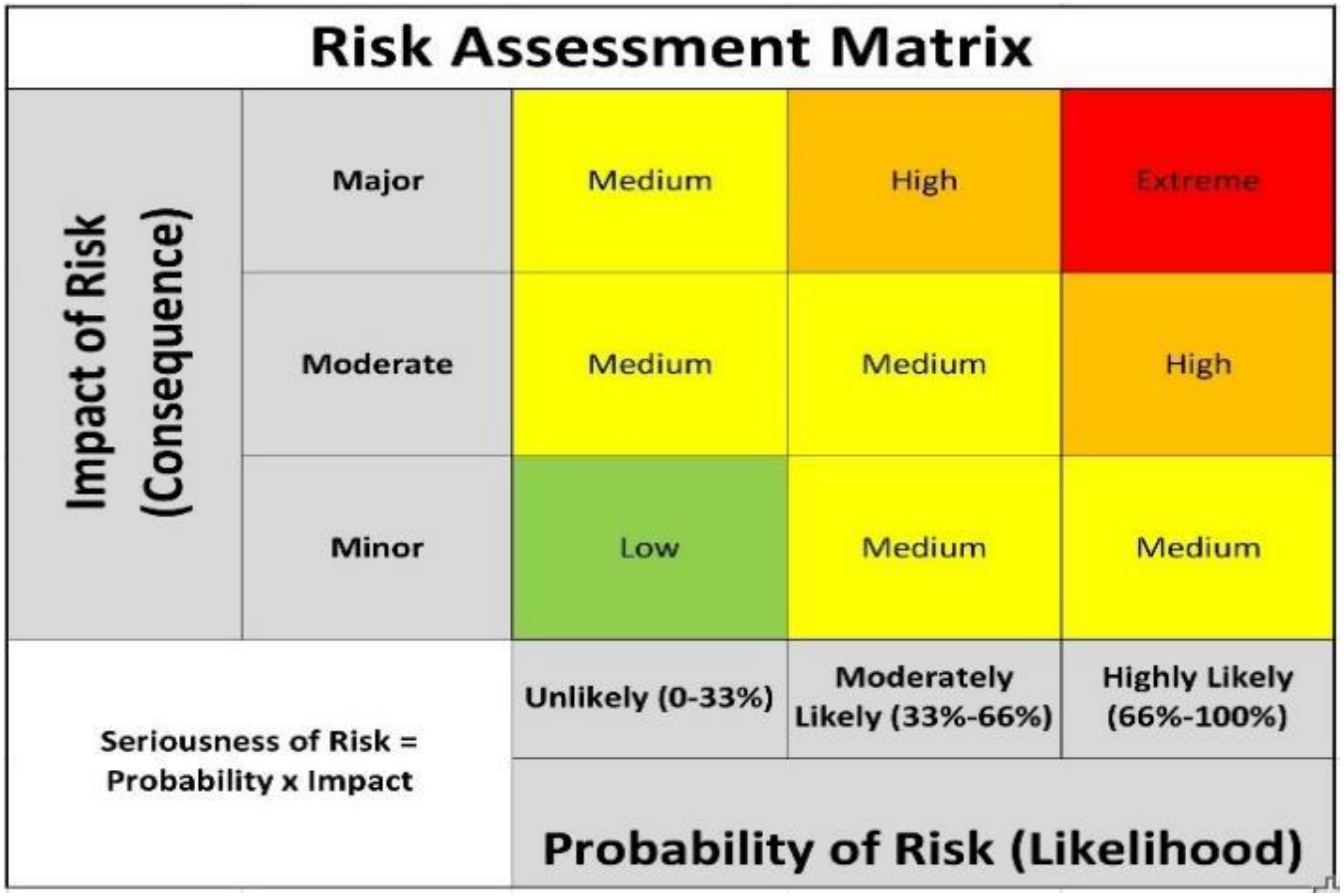
- Assess probability and consequences of each risk.
  - Probability may be:
    - insignificant, low, moderate, high or very high.
  - Risk consequences might be:
    - catastrophic, serious, tolerable or insignificant.

Risk types and examples

HIGH IMPACT !!!

Risk	Probability	Effects
Organizational financial problems force reductions in the project budget (7).	Low	Catastrophic
It is impossible to recruit staff with the skills required for the project (3).	High	Catastrophic
Key staff are ill at critical times in the project (4).	Moderate	Serious
Faults in reusable software components have to be repaired before these components are reused. (2).	Moderate	Serious
Changes to requirements that require major design rework are proposed (10).	Moderate	Serious
The organization is restructured so that different management are responsible for the project (6).	High	Serious
The database used in the system cannot process as many transactions per second as expected (1).	Moderate	Serious
The time required to develop the software is underestimated (12).	High	Serious
Software tools cannot be integrated (9).	High	Tolerable
Customers fail to understand the impact of requirements changes (11).	Moderate	Tolerable
Required training for staff is not available (5).	Moderate	Tolerable
The rate of defect repair is underestimated (13).	Moderate	Tolerable
The size of the software is underestimated (14).	High	Tolerable
Code generated by code generation tools is inefficient (8).	Moderate	Insignificant

Risk Analysis: An Example



Risk Assessment Matrix

Impact of Risk (Consequence)				
	Major	Medium	High	Extreme
	Moderate	Medium	Medium	High
	Minor	Low	Medium	Medium
Seriousness of Risk = Probability x Impact		Unlikely (0-33%)	Moderately Likely (33%-66%)	Highly Likely (66%-100%)
		Probability of Risk (Likelihood)		

Risk planning

- Consider each risk and develop a strategy to manage that risk.
    - Avoidance strategies
        - The probability that the risk will arise is reduced.
    - Minimisation strategies
        - The impact of the risk on the project or product will be reduced.
    - Contingency plans
        - If the risk arises, contingency plans are plans to deal with that risk.

Strategies to help manage risk

Risk	Strategy
Organizational financial problems	Prepare a briefing document for senior management showing how the project is making a very important contribution to the goals of the business and presenting reasons why cuts to the project budget would not be cost-effective.
Recruitment problems	Alert customer to potential difficulties and the possibility of delays; investigate buying-in components.
Staff illness	Reorganize team so that there is more overlap of work and people therefore understand each other's jobs.
Defective components	Replace potentially defective components with bought-in components of known reliability.
Requirements changes	Derive traceability information to assess requirements change impact; maximize information hiding in the design.
Organizational restructuring	Prepare a briefing document for senior management showing how the project is making a very important contribution to the goals of the business.
Database performance	Investigate the possibility of buying a higher-performance database.
Underestimated development time	Investigate buying-in components; investigate use of a program generator.

Risk monitoring

- Assess each identified risks regularly to decide whether or not it is becoming less or more probable.

- Also assess whether the effects of the risk have changed.

- Each key risk should be discussed at management progress meetings.

Risk indicators

Risk type	Potential indicators
Technology	Late delivery of hardware or support software; many reported technology problems.
People	Poor staff morale; poor relationships amongst team members; high staff turnover.
Organizational	Organizational gossip; lack of action by senior management.
Tools	Reluctance by team members to use tools; complaints about CASE tools; demands for higher-powered workstations.
Requirements	Many requirements change requests; customer complaints.
Estimation	Failure to meet agreed schedule; failure to clear reported defects.

Key Points

- Good project management is essential if software engineering projects are to be developed on schedule and within budget.
- Software management is distinct from other engineering management. Software is intangible. Projects may be novel or innovative with no body of experience to guide their management. Software processes are not as mature as traditional engineering processes.
- Risk management is now recognized as one of the most important project management tasks.
- Risk management involves identifying and assessing project risks to establish the probability that they will occur and the consequences for the project if that risk does arise. You should make plans to avoid, manage or deal with likely risks if or when they arise.

CHAPTER 22 – PROJECT PLANNING

Project planning

ORGANIZING
PLANNING
EXECUTION
PROCESS
PROJECT PLANNING
TIMING
CONTROL
ANALYSIS
STRATEGY
MONITORING
DEVELOPMENT

- Project planning involves:
  - breaking down the work into parts,
  - assigning these parts to project team members,
  - anticipating problems that might arise, and
  - preparing tentative solutions to those problems.

- The *project plan*, which is created at the start of a project, is used:
  - to show how the work will be done to the project team and customers, and
  - to help assess progress on the project.

Planning stages

- At the proposal stage,

  → when you are bidding for a contract to develop or provide a software system.

- During the project startup phase,

  → when you have to plan who will work on the project, how the project will be broken down into increments, how resources will be allocated across company, etc.

- Periodically throughout the project,

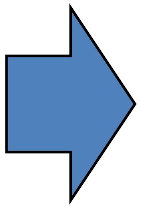
  → when you modify your plan in the light of experience gained and information from monitoring the progress of the work.

Software pricing

- Estimates are made to discover the cost of producing a software system.
  - You take into account: hardware, software, travel, training and effort costs.
- There is <u>not a simple relationship</u> between the development cost and the price charged to the customer.
- Broader organisational, economic, political and business considerations influence the price charged.

Plan-driven development

- *Plan-driven or plan-based development* is an approach to software engineering where the development process is planned in detail.
  - Plan-driven development is based on engineering project management techniques and is the <u>'traditional' way of managing</u> large software development projects.
  - A <u>project plan</u> is created that records the work to be done, who will do it, the development schedule and the work products.
  - Managers use the plan to support project decision making and as a way of measuring progress.



Plan-driven development – pros and cons

- The arguments in favor of a plan-driven approach are that:
  - Early planning allows organizational issues (availability of staff, other projects, etc.) to be closely taken into account, and
  - <u>Potential problems and dependencies are discovered before the project starts</u>, rather than once the project is underway.

- The principal argument against plan-driven development is that:
  - <u>Many early decisions have to be revised</u> because of changes to the environment in which the software is to be developed and used.

Project plans

- In a plan-driven development project, a project plan sets out the resources available to the project, the work breakdown and a schedule for carrying out the work.

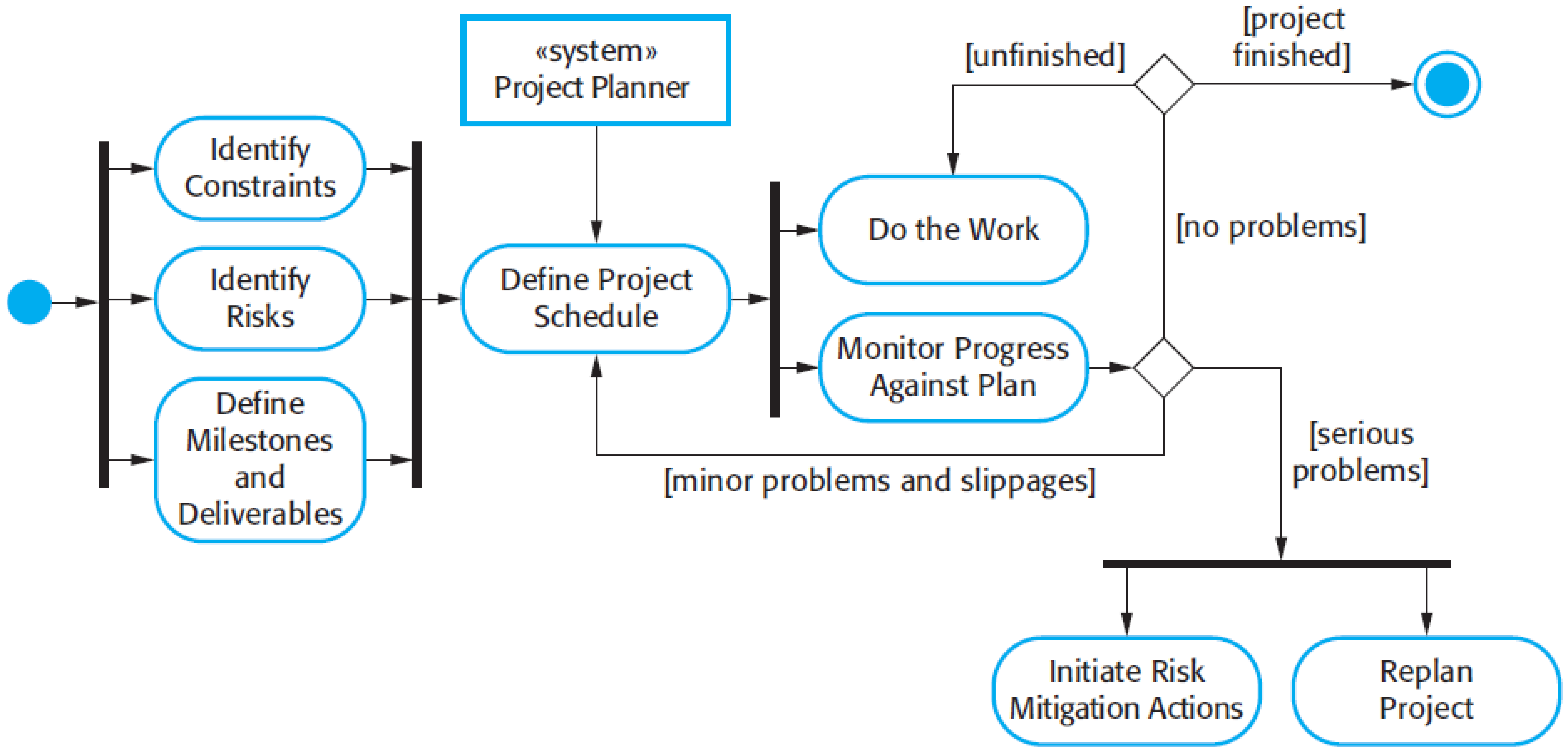
- Plan sections:
  - Introduction
  - Project organization
  - Risk analysis
  - Hardware and software resource requirements
  - Work breakdown
  - Project schedule
  - Monitoring and reporting mechanisms

The planning process

- Project planning is an <u>iterative process</u> that starts when you create an initial project plan during the project startup phase.

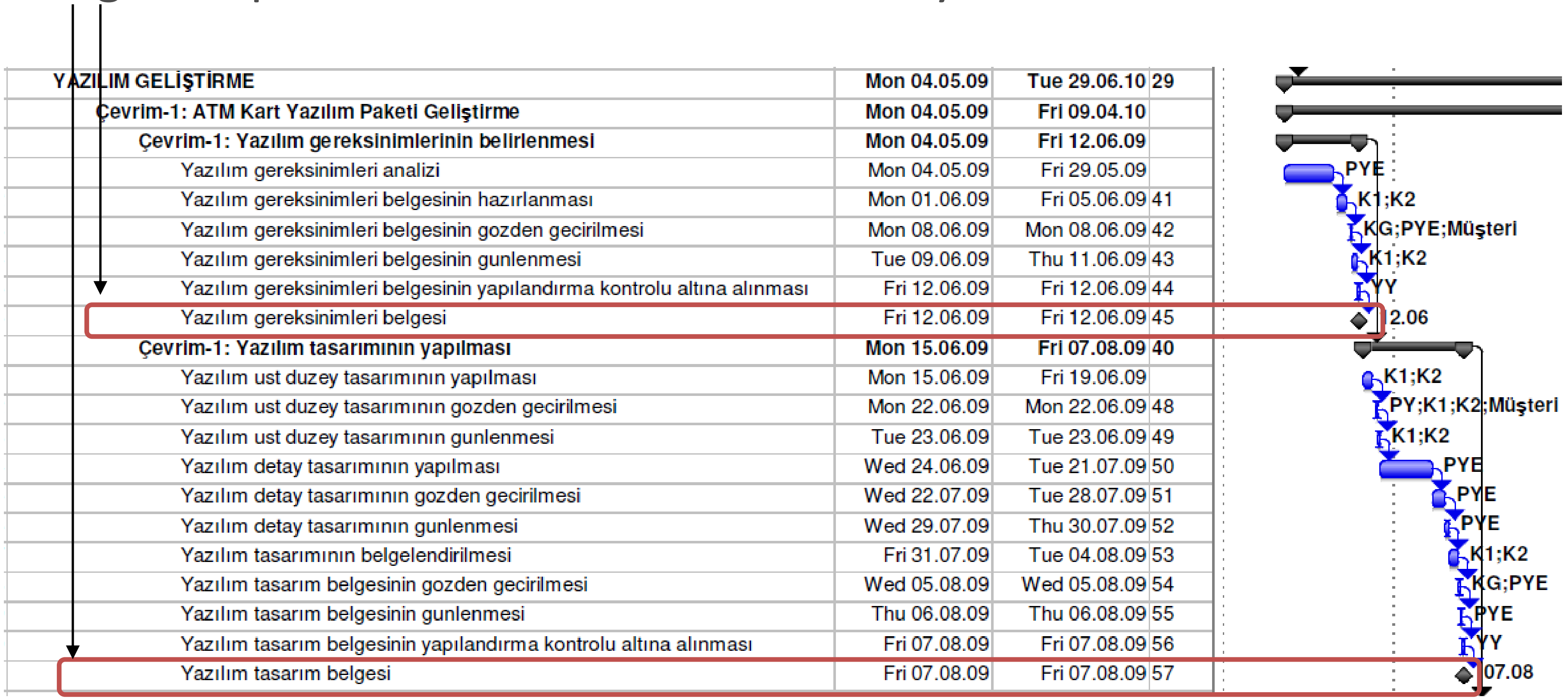
- Plan changes are inevitable.
  - As more information about the system and the project team becomes available during the project, you should regularly revise the plan to reflect requirements, schedule and risk changes.
  - Changing business goals also leads to changes in project plans. As business goals change, this could affect all projects, which may then have to be re-planned.

The project planning process



Milestones and deliverables

- *Milestones* are points in the schedule against which you can assess progress, for example, the handover of the system for testing.
- *Deliverables* are work products that are delivered to the customer, e.g. a requirements document for the system.



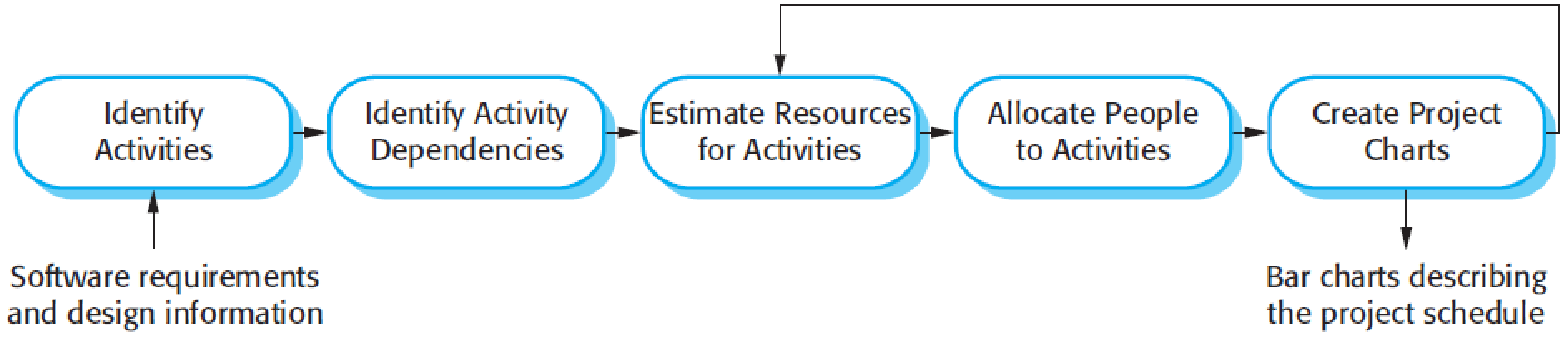
YAZILIM GELİŞTİRME	**Mon 04.05.09**	**Tue 29.06.10**	**29**
Çevrim-1: ATM Kart Yazılım Paketi Geliştirme	**Mon 04.05.09**	**Fri 09.04.10**	
Çevrim-1: Yazılım gereksinimlerinin belirlenmesi	**Mon 04.05.09**	**Fri 12.06.09**	
Yazılım gereksinimleri analizi	Mon 04.05.09	Fri 29.05.09	
Yazılım gereksinimleri belgesinin hazırlanması	Mon 01.06.09	Fri 05.06.09	41
Yazılım gereksinimleri belgesinin gozden gecirilmesi	Mon 08.06.09	Mon 08.06.09	42
Yazılım gereksinimleri belgesinin gunlenmesi	Tue 09.06.09	Thu 11.06.09	43
Yazılım gereksinimleri belgesinin yapılandırma kontrolu altına alınması	Fri 12.06.09	Fri 12.06.09	44
Yazılım gereksinimleri belgesi	Fri 12.06.09	Fri 12.06.09	45
Çevrim-1: Yazılım tasarımının yapılması	**Mon 15.06.09**	**Fri 07.08.09**	**40**
Yazılım ust duzey tasarımının yapılması	Mon 15.06.09	Fri 19.06.09	
Yazılım ust duzey tasarımının gozden gecirilmesi	Mon 22.06.09	Mon 22.06.09	48
Yazılım ust duzey tasarımının gunlenmesi	Tue 23.06.09	Tue 23.06.09	49
Yazılım detay tasarımının yapılması	Wed 24.06.09	Tue 21.07.09	50
Yazılım detay tasarımının gozden gecirilmesi	Wed 22.07.09	Tue 28.07.09	51
Yazılım detay tasarımının gunlenmesi	Wed 29.07.09	Thu 30.07.09	52
Yazılım tasarımının belgelendirilmesi	Fri 31.07.09	Tue 04.08.09	53
Yazılım tasarım belgesinin gozden gecirilmesi	Wed 05.08.09	Wed 05.08.09	54
Yazılım tasarım belgesinin gunlenmesi	Thu 06.08.09	Thu 06.08.09	55
Yazılım tasarım belgesinin yapılandırma kontrolu altına alınması	Fri 07.08.09	Fri 07.08.09	56
Yazılım tasarım belgesi	Fri 07.08.09	Fri 07.08.09	57

Project scheduling

- *Project scheduling* is the process of deciding how the work in a project will be organized as separate tasks, and when and how these tasks will be executed.
- You estimate the calendar time needed to complete each task, the effort required and who will work on the tasks that have been identified.
- You also have to estimate the resources needed to complete each task, such as the disk space required on a server, the time required on specialized hardware, such as a simulator, and what the travel budget will be.

Project scheduling activities

- Split project into tasks and estimate time and resources required to complete each task.
- Organize tasks concurrently to make optimal use of workforce.
- Minimize task dependencies to avoid delays caused by one task waiting for another to complete.
- Dependent on project managers intuition and experience.


Identify Activities
Identify Activity Dependencies
Estimate Resources for Activities
Allocate People to Activities
Create Project Charts
Software requirements and design information
Bar charts describing the project schedule

Scheduling problems

- Estimating the difficulty of problems and hence the cost of developing a solution is hard.
- Productivity is not proportional to the number of people working on a task.
- <u>Adding people to a late project makes it later</u> because of communication overheads.
- The unexpected always happens. Always allow contingency in planning.

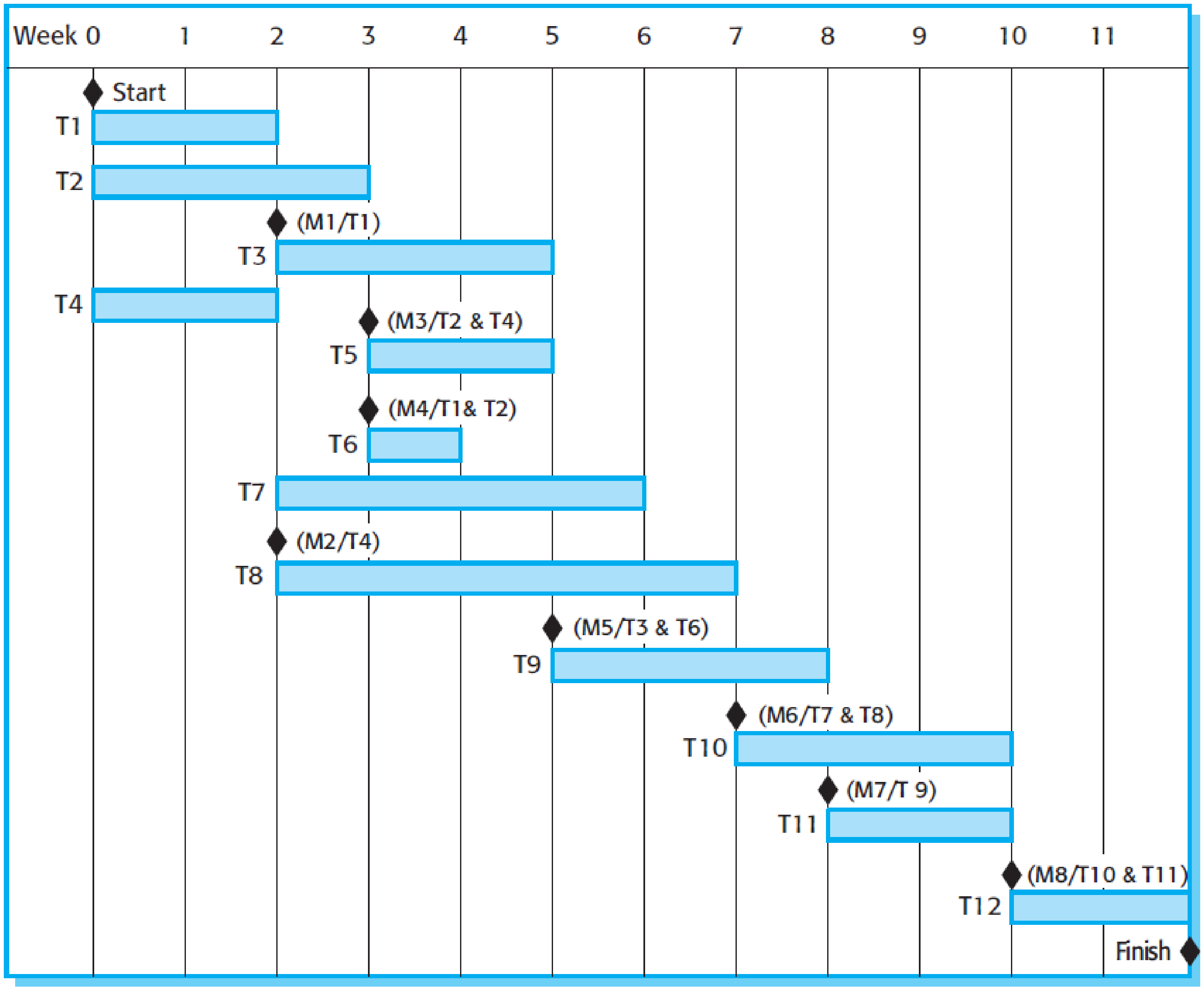
Schedule representation

- Graphical notations are normally used to illustrate the project schedule.
- These show the project breakdown into tasks. Tasks should not be too small. They should take about *a week or two*.
- Bar charts are the most commonly used representation for project schedules. They show the schedule as activities or resources against time.

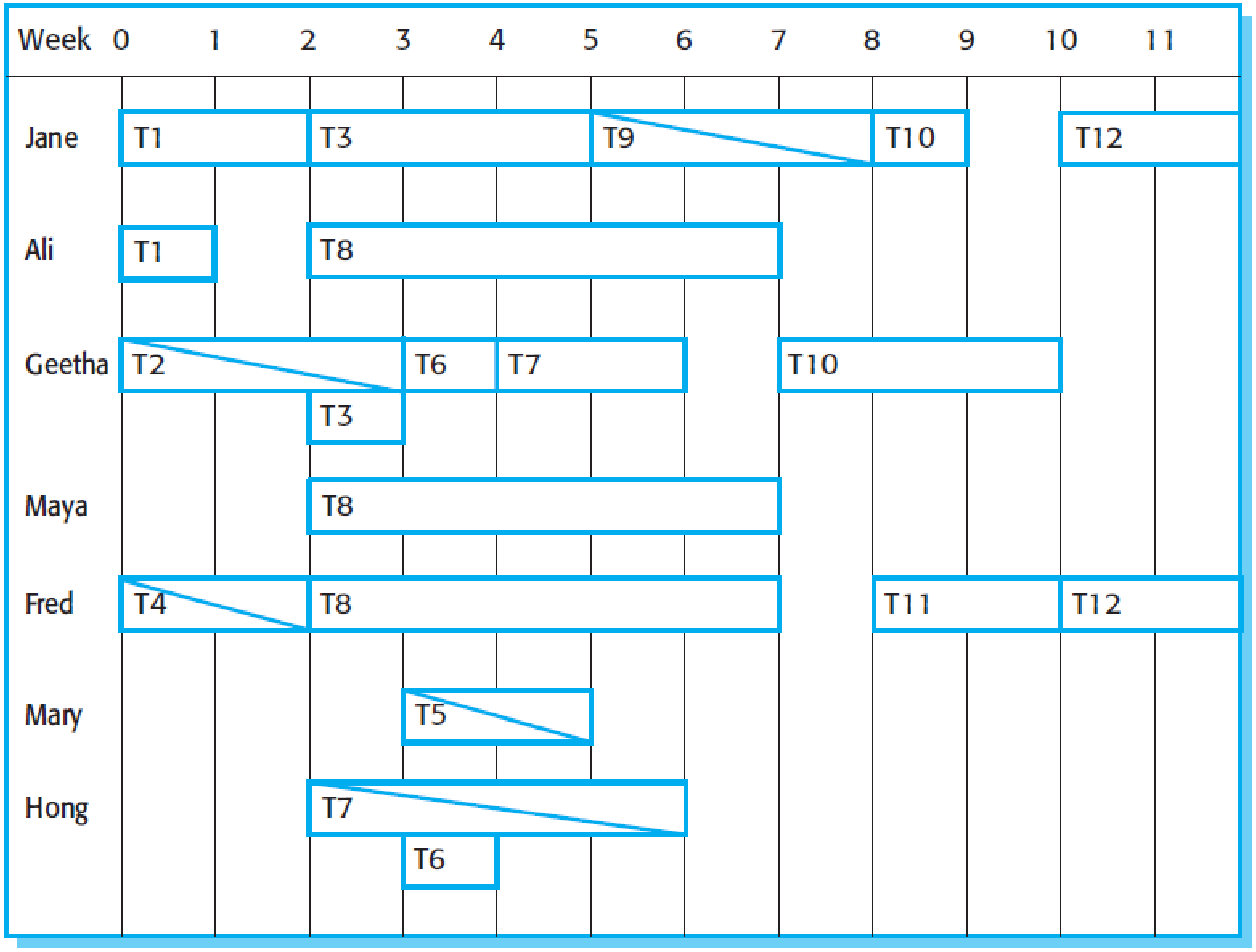
Tasks, durations, and dependencies

Task	Effort (person-days)	Duration (days)	Dependencies
T1	15	10	
T2	8	15	
T3	20	15	T1 (M1)
T4	5	10	
T5	5	10	T2, T4 (M3)
T6	10	5	T1, T2 (M4)
T7	25	20	T1 (M1)
T8	75	25	T4 (M2)
T9	10	15	T3, T6 (M5)
T10	20	15	T7, T8 (M6)
T11	10	10	T9 (M7)
T12	20	10	T10, T11 (M8)

Activity bar chart



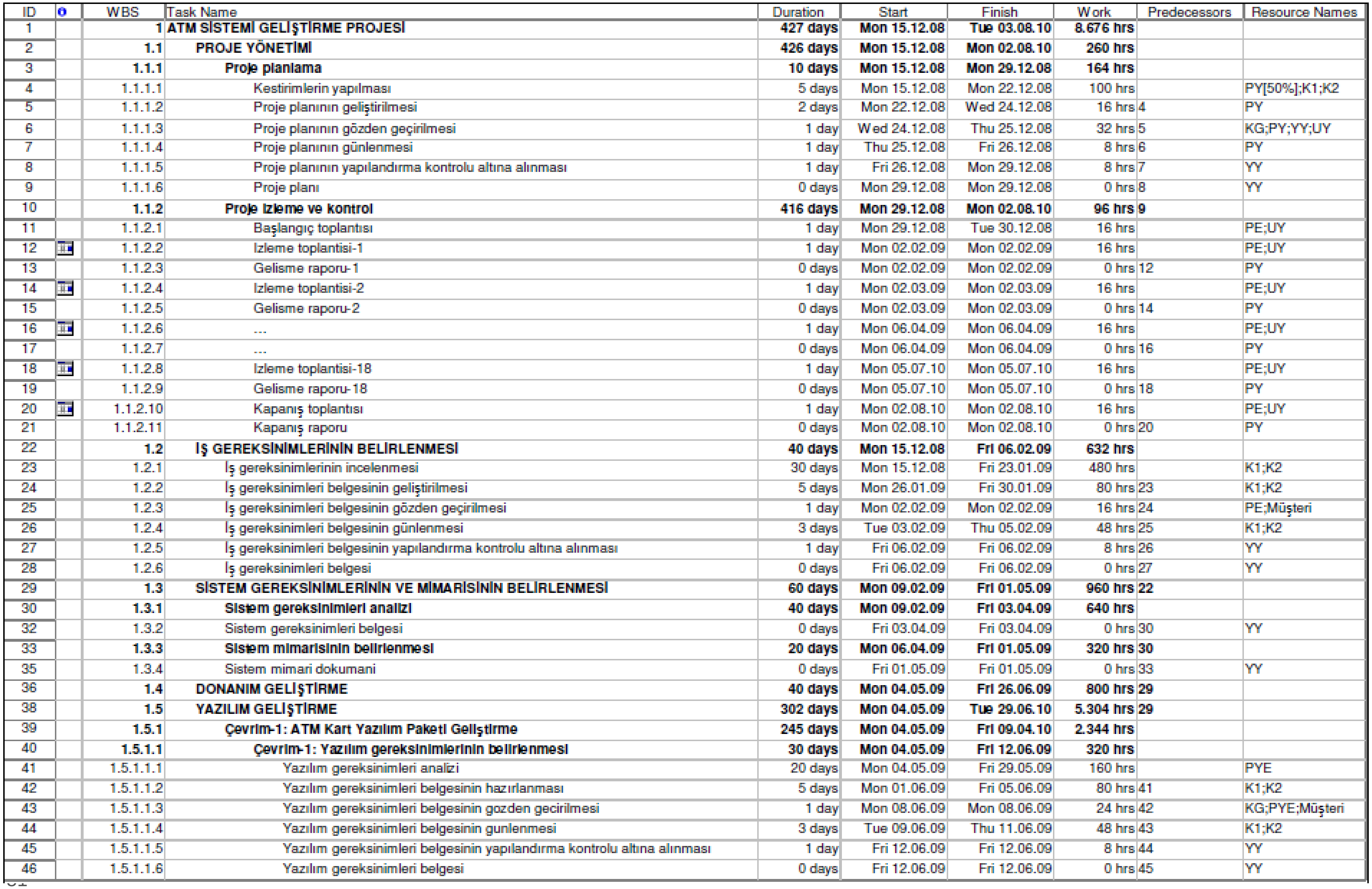
Staff allocation chart



Example:
ATM System Software Development Project Plan

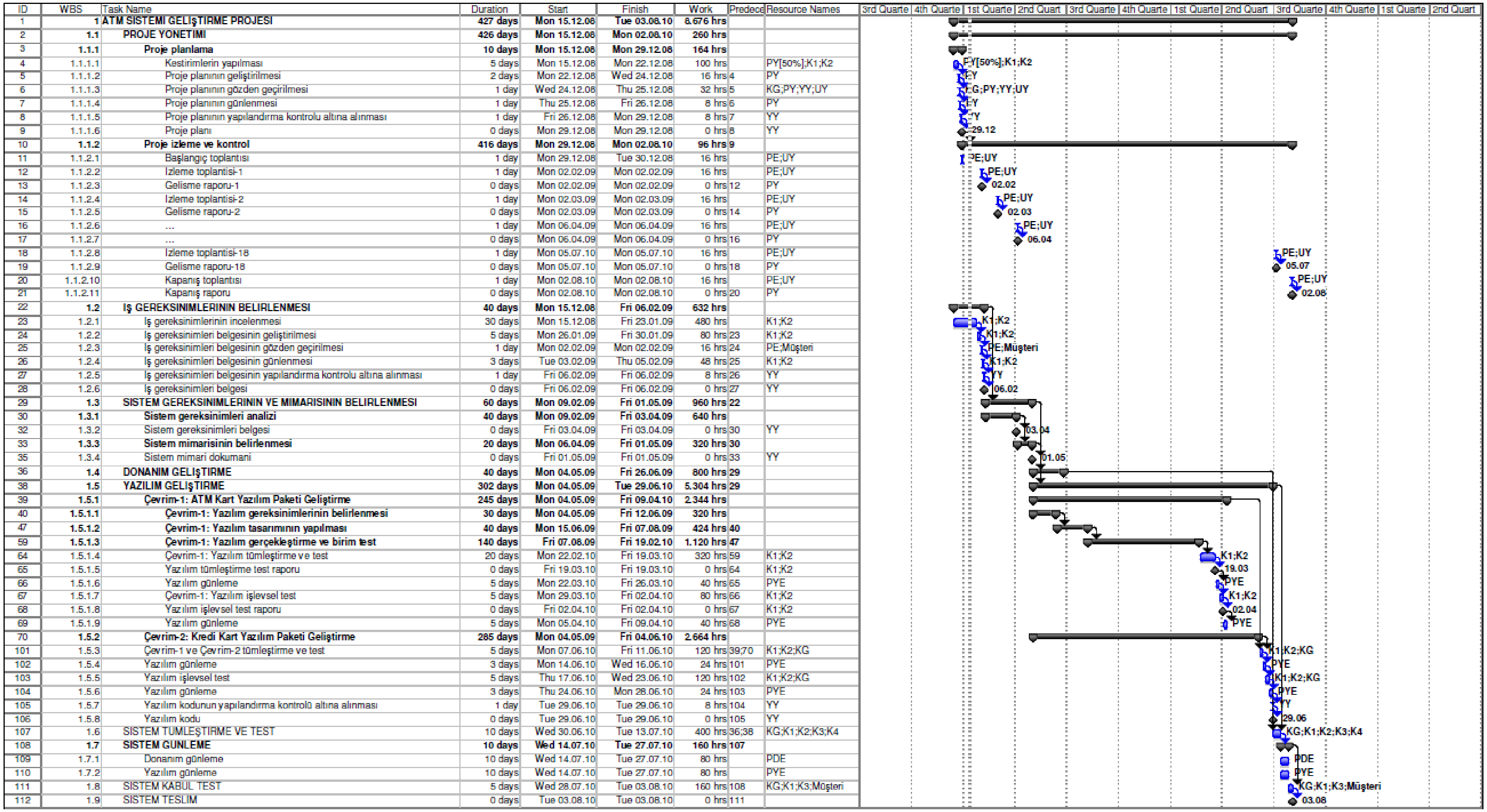
- Full plan is available in the share area of course lecture notes.

Example: ATM System Software Development (Tasks and dependencies, Schedule, Effort, Resources)



ID	●	WBS	Task Name	Duration	Start	Finish	Work	Predecessors	Resource Names
1		1	**ATM SİSTEMİ GELİŞTİRME PROJESİ**	**427 days**	**Mon 15.12.08**	**Tue 03.08.10**	**8.676 hrs**		
2		**1.1**	**PROJE YÖNETİMİ**	**426 days**	**Mon 15.12.08**	**Mon 02.08.10**	**260 hrs**		
3		**1.1.1**	**Proje planlama**	**10 days**	**Mon 15.12.08**	**Mon 29.12.08**	**164 hrs**		
4		1.1.1.1	Kestirimlerin yapılması	5 days	Mon 15.12.08	Mon 22.12.08	100 hrs		PY[50%];K1;K2
5		1.1.1.2	Proje planının geliştirilmesi	2 days	Mon 22.12.08	Wed 24.12.08	16 hrs	4	PY
6		1.1.1.3	Proje planının gözden geçirilmesi	1 day	Wed 24.12.08	Thu 25.12.08	32 hrs	5	KG;PY;YY;UY
7		1.1.1.4	Proje planının günlenmesi	1 day	Thu 25.12.08	Fri 26.12.08	8 hrs	6	PY
8		1.1.1.5	Proje planının yapılandırma kontrolu altına alınması	1 day	Fri 26.12.08	Mon 29.12.08	8 hrs	7	YY
9		1.1.1.6	Proje planı	0 days	Mon 29.12.08	Mon 29.12.08	0 hrs	8	YY
10		**1.1.2**	**Proje izleme ve kontrol**	**416 days**	**Mon 29.12.08**	**Mon 02.08.10**	**96 hrs**	**9**	
11		1.1.2.1	Başlangıç toplantısı	1 day	Mon 29.12.08	Tue 30.12.08	16 hrs		PE;UY
12	▦	1.1.2.2	Izleme toplantisi-1	1 day	Mon 02.02.09	Mon 02.02.09	16 hrs		PE;UY
13		1.1.2.3	Gelisme raporu-1	0 days	Mon 02.02.09	Mon 02.02.09	0 hrs	12	PY
14	▦	1.1.2.4	Izleme toplantisi-2	1 day	Mon 02.03.09	Mon 02.03.09	16 hrs		PE;UY
15		1.1.2.5	Gelisme raporu-2	0 days	Mon 02.03.09	Mon 02.03.09	0 hrs	14	PY
16	▦	1.1.2.6	...	1 day	Mon 06.04.09	Mon 06.04.09	16 hrs		PE;UY
17		1.1.2.7	...	0 days	Mon 06.04.09	Mon 06.04.09	0 hrs	16	PY
18	▦	1.1.2.8	Izleme toplantisi-18	1 day	Mon 05.07.10	Mon 05.07.10	16 hrs		PE;UY
19		1.1.2.9	Gelisme raporu-18	0 days	Mon 05.07.10	Mon 05.07.10	0 hrs	18	PY
20	▦	1.1.2.10	Kapanış toplantısı	1 day	Mon 02.08.10	Mon 02.08.10	16 hrs		PE;UY
21		1.1.2.11	Kapanış raporu	0 days	Mon 02.08.10	Mon 02.08.10	0 hrs	20	PY
22		**1.2**	**İŞ GEREKSİNİMLERİNİN BELİRLENMESİ**	**40 days**	**Mon 15.12.08**	**Fri 06.02.09**	**632 hrs**		
23		1.2.1	İş gereksinimlerinin incelenmesi	30 days	Mon 15.12.08	Fri 23.01.09	480 hrs		K1;K2
24		1.2.2	İş gereksinimleri belgesinin geliştirilmesi	5 days	Mon 26.01.09	Fri 30.01.09	80 hrs	23	K1;K2
25		1.2.3	İş gereksinimleri belgesinin gözden geçirilmesi	1 day	Mon 02.02.09	Mon 02.02.09	16 hrs	24	PE;Müşteri
26		1.2.4	İş gereksinimleri belgesinin günlenmesi	3 days	Tue 03.02.09	Thu 05.02.09	48 hrs	25	K1;K2
27		1.2.5	İş gereksinimleri belgesinin yapılandırma kontrolu altına alınması	1 day	Fri 06.02.09	Fri 06.02.09	8 hrs	26	YY
28		1.2.6	İş gereksinimleri belgesi	0 days	Fri 06.02.09	Fri 06.02.09	0 hrs	27	YY
29		**1.3**	**SİSTEM GEREKSİNİMLERİNİN VE MİMARİSİNİN BELİRLENMESİ**	**60 days**	**Mon 09.02.09**	**Fri 01.05.09**	**960 hrs**	**22**	
30		**1.3.1**	**Sistem gereksinimleri analizi**	**40 days**	**Mon 09.02.09**	**Fri 03.04.09**	**640 hrs**		
32		1.3.2	Sistem gereksinimleri belgesi	0 days	Fri 03.04.09	Fri 03.04.09	0 hrs	30	YY
33		**1.3.3**	**Sistem mimarisinin belirlenmesi**	**20 days**	**Mon 06.04.09**	**Fri 01.05.09**	**320 hrs**	**30**	
35		1.3.4	Sistem mimari dokumani	0 days	Fri 01.05.09	Fri 01.05.09	0 hrs	33	YY
36		**1.4**	**DONANIM GELİŞTİRME**	**40 days**	**Mon 04.05.09**	**Fri 26.06.09**	**800 hrs**	**29**	
38		**1.5**	**YAZILIM GELİŞTİRME**	**302 days**	**Mon 04.05.09**	**Tue 29.06.10**	**5.304 hrs**	**29**	
39		**1.5.1**	**Çevrim-1: ATM Kart Yazılım Paketi Geliştirme**	**245 days**	**Mon 04.05.09**	**Fri 09.04.10**	**2.344 hrs**		
40		**1.5.1.1**	**Çevrim-1: Yazılım gereksinimlerinin belirlenmesi**	**30 days**	**Mon 04.05.09**	**Fri 12.06.09**	**320 hrs**		
41		1.5.1.1.1	Yazılım gereksinimleri analizi	20 days	Mon 04.05.09	Fri 29.05.09	160 hrs		PYE
42		1.5.1.1.2	Yazılım gereksinimleri belgesinin hazırlanması	5 days	Mon 01.06.09	Fri 05.06.09	80 hrs	41	K1;K2
43		1.5.1.1.3	Yazılım gereksinimleri belgesinin gozden gecirilmesi	1 day	Mon 08.06.09	Mon 08.06.09	24 hrs	42	KG;PYE;Müşteri
44		1.5.1.1.4	Yazılım gereksinimleri belgesinin gunlenmesi	3 days	Tue 09.06.09	Thu 11.06.09	48 hrs	43	K1;K2
45		1.5.1.1.5	Yazılım gereksinimleri belgesinin yapılandırma kontrolu altına alınması	1 day	Fri 12.06.09	Fri 12.06.09	8 hrs	44	YY
46		1.5.1.1.6	Yazılım gereksinimleri belgesi	0 days	Fri 12.06.09	Fri 12.06.09	0 hrs	45	YY

Example: ATM System Software Development
(Tasks and dependencies – Gantt Chart)



ID	WBS	Task Name	Duration	Start	Finish	Work	Predece	Resource Names
1	1	**ATM SISTEMI GELIşTIRME PROJESI**	**427 days**	**Mon 15.12.08**	**Tue 03.08.10**	**8.676 hrs**		
2	**1.1**	**PROJE YONETIMI**	**426 days**	**Mon 15.12.08**	**Mon 02.08.10**	**260 hrs**		
3	**1.1.1**	**Proje planlama**	**10 days**	**Mon 15.12.08**	**Mon 29.12.08**	**164 hrs**		
4	1.1.1.1	Kestirimlerin yapılması	5 days	Mon 15.12.08	Mon 22.12.08	100 hrs		PY[50%];K1;K2
5	1.1.1.2	Proje planının geliştirilmesi	2 days	Mon 22.12.08	Wed 24.12.08	16 hrs	4	PY
6	1.1.1.3	Proje planının gözden geçirilmesi	1 day	Wed 24.12.08	Thu 25.12.08	32 hrs	5	KG;PY;YY;UY
7	1.1.1.4	Proje planının günlenmesi	1 day	Thu 25.12.08	Fri 26.12.08	8 hrs	6	PY
8	1.1.1.5	Proje planının yapılandırma kontrolu altına alınması	1 day	Fri 26.12.08	Mon 29.12.08	8 hrs	7	YY
9	1.1.1.6	Proje planı	0 days	Mon 29.12.08	Mon 29.12.08	0 hrs	8	YY
10	**1.1.2**	**Proje izleme ve kontrol**	**416 days**	**Mon 29.12.08**	**Mon 02.08.10**	**96 hrs**	**9**	
11	1.1.2.1	Başlangıç toplantısı	1 day	Mon 29.12.08	Tue 30.12.08	16 hrs		PE;UY
12	1.1.2.2	Izleme toplantisi-1	1 day	Mon 02.02.09	Mon 02.02.09	16 hrs		PE;UY
13	1.1.2.3	Gelisme raporu-1	0 days	Mon 02.02.09	Mon 02.02.09	0 hrs	12	PY
14	1.1.2.4	Izleme toplantisi-2	1 day	Mon 02.03.09	Mon 02.03.09	16 hrs		PE;UY
15	1.1.2.5	Gelisme raporu-2	0 days	Mon 02.03.09	Mon 02.03.09	0 hrs	14	PY
16	1.1.2.6	...	1 day	Mon 06.04.09	Mon 06.04.09	16 hrs		PE;UY
17	1.1.2.7	...	0 days	Mon 06.04.09	Mon 06.04.09	0 hrs	16	PY
18	1.1.2.8	Izleme toplantisi-18	1 day	Mon 05.07.10	Mon 05.07.10	16 hrs		PE;UY
19	1.1.2.9	Gelisme raporu-18	0 days	Mon 05.07.10	Mon 05.07.10	0 hrs	18	PY
20	1.1.2.10	Kapanış toplantısı	1 day	Mon 02.08.10	Mon 02.08.10	16 hrs		PE;UY
21	1.1.2.11	Kapanış raporu	0 days	Mon 02.08.10	Mon 02.08.10	0 hrs	20	PY
22	**1.2**	**Iş GEREKSINIMLERININ BELIRLENMESI**	**40 days**	**Mon 15.12.08**	**Fri 06.02.09**	**632 hrs**		
23	1.2.1	Iş gereksinimlerinin incelenmesi	30 days	Mon 15.12.08	Fri 23.01.09	480 hrs		K1;K2
24	1.2.2	Iş gereksinimleri belgesinin geliştirilmesi	5 days	Mon 26.01.09	Fri 30.01.09	80 hrs	23	K1;K2
25	1.2.3	Iş gereksinimleri belgesinin gözden geçirilmesi	1 day	Mon 02.02.09	Mon 02.02.09	16 hrs	24	PE;Müşteri
26	1.2.4	Iş gereksinimleri belgesinin günlenmesi	3 days	Tue 03.02.09	Thu 05.02.09	48 hrs	25	K1;K2
27	1.2.5	Iş gereksinimleri belgesinin yapılandırma kontrolu altına alınması	1 day	Fri 06.02.09	Fri 06.02.09	8 hrs	26	YY
28	1.2.6	Iş gereksinimleri belgesi	0 days	Fri 06.02.09	Fri 06.02.09	0 hrs	27	YY
29	**1.3**	**SISTEM GEREKSINIMLERININ VE MIMARISININ BELIRLENMESI**	**60 days**	**Mon 09.02.09**	**Fri 01.05.09**	**960 hrs**	**22**	
30	**1.3.1**	**Sistem gereksinimleri analizi**	**40 days**	**Mon 09.02.09**	**Fri 03.04.09**	**640 hrs**		
32	1.3.2	Sistem gereksinimleri belgesi	0 days	Fri 03.04.09	Fri 03.04.09	0 hrs	30	YY
33	**1.3.3**	**Sistem mimarisinin belirlenmesi**	**20 days**	**Mon 06.04.09**	**Fri 01.05.09**	**320 hrs**	**30**	
35	1.3.4	Sistem mimari dokumani	0 days	Fri 01.05.09	Fri 01.05.09	0 hrs	33	YY
36	**1.4**	**DONANIM GELIşTIRME**	**40 days**	**Mon 04.05.09**	**Fri 26.06.09**	**800 hrs**	**29**	
38	**1.5**	**YAZILIM GELIşTIRME**	**302 days**	**Mon 04.05.09**	**Tue 29.06.10**	**5.304 hrs**	**29**	
39	**1.5.1**	**Çevrim-1: ATM Kart Yazılım Paketi Geliştirme**	**245 days**	**Mon 04.05.09**	**Fri 09.04.10**	**2.344 hrs**		
40	**1.5.1.1**	**Çevrim-1: Yazılım gereksinimlerinin belirlenmesi**	**30 days**	**Mon 04.05.09**	**Fri 12.06.09**	**320 hrs**		
47	**1.5.1.2**	**Çevrim-1: Yazılım tasarımının yapılması**	**40 days**	**Mon 15.06.09**	**Fri 07.08.09**	**424 hrs**	**40**	
59	**1.5.1.3**	**Çevrim-1: Yazılım gerçekleştirme ve birim test**	**140 days**	**Fri 07.08.09**	**Fri 19.02.10**	**1.120 hrs**	**47**	
64	1.5.1.4	Çevrim-1: Yazılım tümleştirme ve test	20 days	Mon 22.02.10	Fri 19.03.10	320 hrs	59	K1;K2
65	1.5.1.5	Yazılım tümleştirme test raporu	0 days	Fri 19.03.10	Fri 19.03.10	0 hrs	64	K1;K2
66	1.5.1.6	Yazılım günleme	5 days	Mon 22.03.10	Fri 26.03.10	40 hrs	65	PYE
67	1.5.1.7	Çevrim-1: Yazılım işlevsel test	5 days	Mon 29.03.10	Fri 02.04.10	80 hrs	66	K1;K2
68	1.5.1.8	Yazılım işlevsel test raporu	0 days	Fri 02.04.10	Fri 02.04.10	0 hrs	67	K1;K2
69	1.5.1.9	Yazılım günleme	5 days	Mon 05.04.10	Fri 09.04.10	40 hrs	68	PYE
70	**1.5.2**	**Çevrim-2: Kredi Kart Yazılım Paketi Geliştirme**	**285 days**	**Mon 04.05.09**	**Fri 04.06.10**	**2.664 hrs**		
101	1.5.3	Çevrim-1 ve Çevrim-2 tümleştirme ve test	5 days	Mon 07.06.10	Fri 11.06.10	120 hrs	39;70	K1;K2;KG
102	1.5.4	Yazılım günleme	3 days	Mon 14.06.10	Wed 16.06.10	24 hrs	101	PYE
103	1.5.5	Yazılım işlevsel test	5 days	Thu 17.06.10	Wed 23.06.10	120 hrs	102	K1;K2;KG
104	1.5.6	Yazılım günleme	3 days	Thu 24.06.10	Mon 28.06.10	24 hrs	103	PYE
105	1.5.7	Yazılım kodunun yapılandırma kontrolü altına alınması	1 day	Tue 29.06.10	Tue 29.06.10	8 hrs	104	YY
106	1.5.8	Yazılım kodu	0 days	Tue 29.06.10	Tue 29.06.10	0 hrs	105	YY
107	1.6	SISTEM TUMLEŞTIRME VE TEST	10 days	Wed 30.06.10	Tue 13.07.10	400 hrs	36;38	KG;K1;K2;K3;K4
108	**1.7**	**SISTEM GUNLEME**	**10 days**	**Wed 14.07.10**	**Tue 27.07.10**	**160 hrs**	**107**	
109	1.7.1	Donanım günleme	10 days	Wed 14.07.10	Tue 27.07.10	80 hrs		PDE
110	1.7.2	Yazılım günleme	10 days	Wed 14.07.10	Tue 27.07.10	80 hrs		PYE
111	1.8	SISTEM KABUL TEST	5 days	Wed 28.07.10	Tue 03.08.10	160 hrs	108	KG;K1;K3;Müşteri
112	1.9	SISTEM TESLIM	0 days	Tue 03.08.10	Tue 03.08.10	0 hrs	111	

Key points

- The price charged for a system does not just depend on its estimated development costs; it may be adjusted depending on the market and organizational priorities.
- Plan-driven development is organized around a complete project plan that defines the project activities, the planned effort, the activity schedule and who is responsible for each activity.
- Project scheduling involves the creation of graphical representations the project plan. Bar charts show the activity duration and staffing timelines, are the most commonly used schedule representations.

FOR STUDENT READING…

Project plan supplements

Plan	Description
Quality plan	Describes the quality procedures and standards that will be used in a project.
Validation plan	Describes the approach, resources, and schedule used for system validation.
Configuration management plan	Describes the configuration management procedures and structures to be used.
Maintenance plan	Predicts the maintenance requirements, costs, and effort.
Staff development plan	Describes how the skills and experience of the project team members will be developed.

Factors affecting software pricing

Factor	Description
Market opportunity	A development organization 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 organization the opportunity to make a greater profit later. The experience gained may also help it develop new products.
Cost estimate uncertainty	If an organization is unsure of its cost estimate, it may increase its price by a 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 organization may lower its price to win a contract. After the contract is awarded, high prices can 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 smaller than normal profit or break even than to go out of business. Cash flow is more important than profit in difficult economic times.

Estimation techniques

- Organizations need to make software effort and cost estimates. There are two types of technique that can be used to do this:
  - *Experience-based techniques* The estimate of future effort requirements is based on the manager's experience of past projects and the application domain. Essentially, the manager makes an informed judgment of what the effort requirements are likely to be.
  - *Algorithmic cost modeling* In this approach, a formulaic approach is used to compute the project effort based on estimates of product attributes, such as size, and process characteristics, such as experience of staff involved.

Experience-based approaches

- Experience-based techniques rely on judgments based on experience of past projects and the effort expended in these projects on software development activities.
- Typically, you identify the deliverables to be produced in a project and the different software components or systems that are to be developed.
- You document these in a spreadsheet, estimate them individually and compute the total effort required.
- It usually helps to get a group of people involved in the effort estimation and to ask each member of the group to explain their estimate.

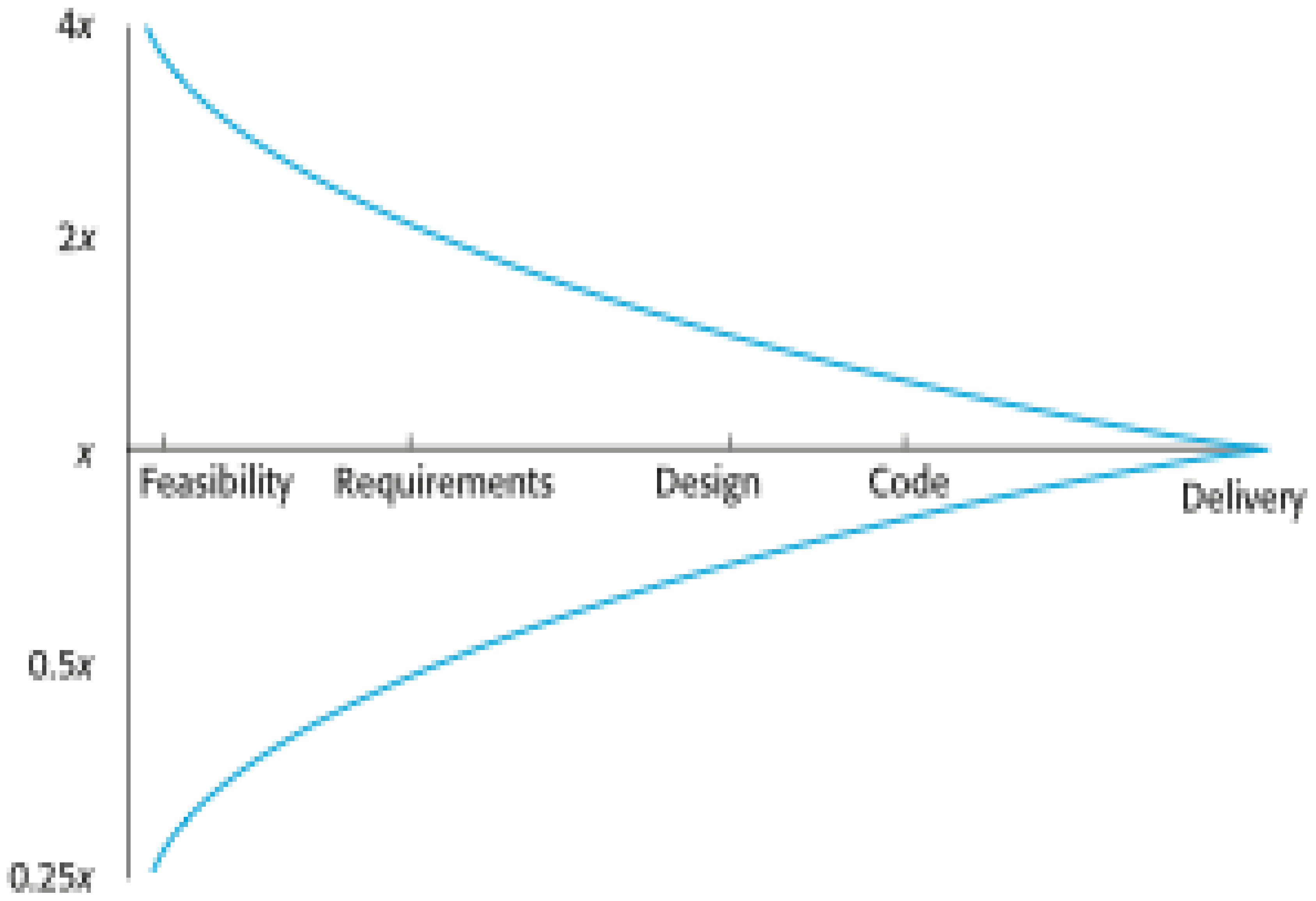
Algorithmic cost modelling

- Cost is estimated as a mathematical function of product, project and process attributes whose values are estimated by project managers:
  - Effort = A ´ Size^{B} ´ M
  - A is an organisation-dependent constant, B reflects the disproportionate effort for large projects and M is a multiplier reflecting product, process and people attributes.
- The most commonly used product attribute for cost estimation is code size.
- Most models are similar but they use different values for A, B and M.

Estimation accuracy

- The 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 size estimate becomes more accurate.
- The estimates of the factors contributing to B and M are subjective and vary according to the judgment of the estimator.

Estimate uncertainty



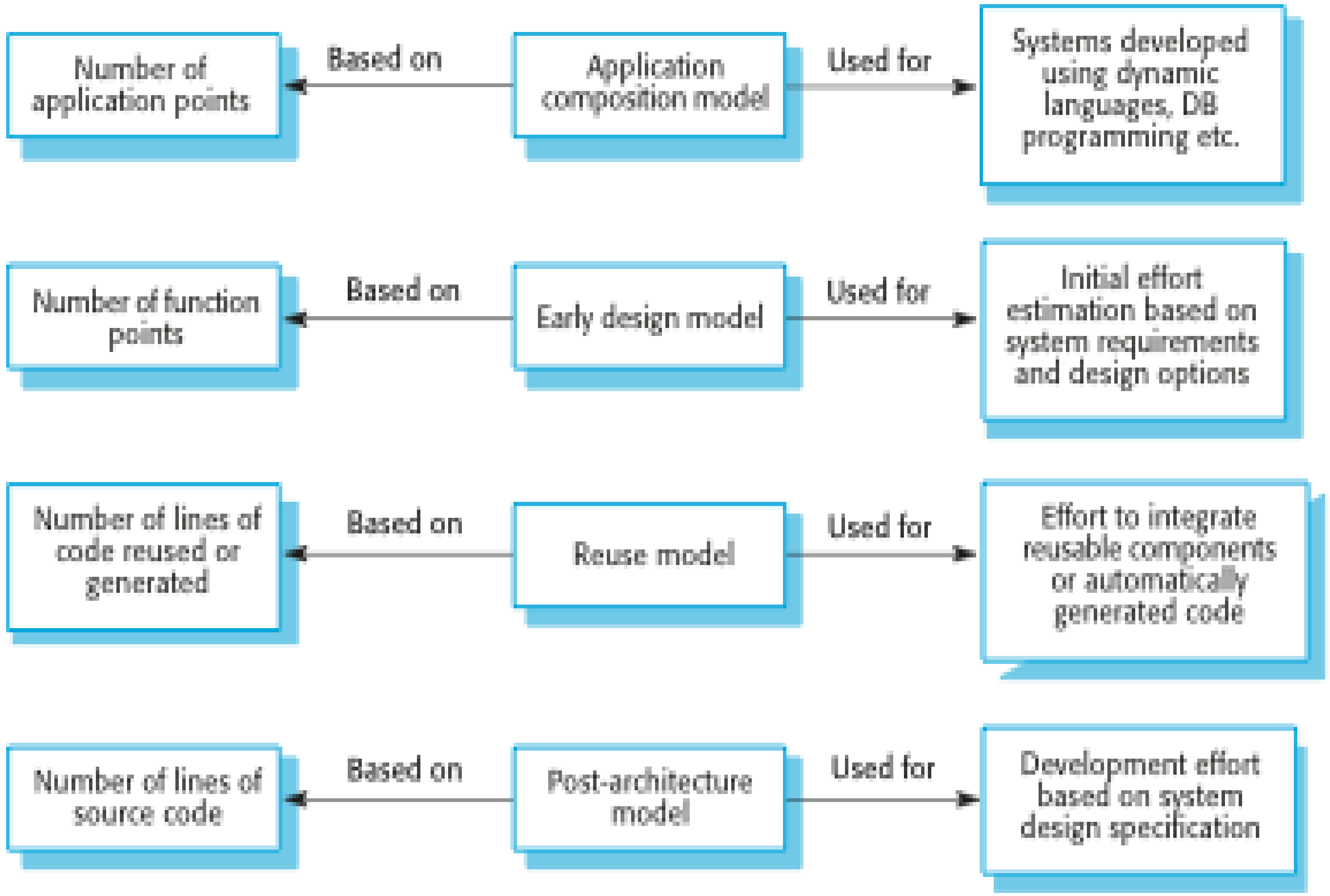
The COCOMO 2 model

- An empirical model based on project experience.
- Well-documented, ‘independent’ model which is not tied to a specific software vendor.
- Long history from initial version published in 1981 (COCOMO-81) through various instantiations to COCOMO 2.
- COCOMO 2 takes into account different approaches to software development, reuse, etc.

COCOMO 2 models

- COCOMO 2 incorporates a range of sub-models that produce increasingly detailed software estimates.
- The sub-models in COCOMO 2 are:
  - Application composition model. Used when software is composed from existing parts.
  - Early design model. Used when requirements are available but design has not yet started.
  - Reuse model. Used to compute the effort of integrating reusable components.
  - Post-architecture model. Used once the system architecture has been designed and more information about the system is available.

COCOMO estimation models


Number of application points
Based on
Application composition model
Used for
Systems developed using dynamic languages, DB programming etc.
Number of function points
Based on
Early design model
Used for
Initial effort estimation based on system requirements and design options
Number of lines of code reused or generated
Based on
Reuse model
Used for
Effort to integrate reusable components or automatically generated code
Number of lines of source code
Based on
Post-architecture model
Used for
Development effort based on system design specification

Application composition model

- Supports prototyping projects and projects where there is extensive reuse.
- Based on standard estimates of developer productivity in application (object) points/month.
- Takes CASE tool use into account.
- Formula is
  - PM = (NAP ´ (1 - %reuse/100)) / PROD
  - PM is the effort in person-months, NAP is the number of application points and PROD is the productivity.

Application-point productivity

Developer's experience and capability	Very low	Low	Nominal	High	Very high
ICASE maturity and capability	Very low	Low	Nominal	High	Very high
PROD (NAP/month)	4	7	13	25	50

Early design model

- Estimates can be made after the requirements have been agreed.
- Based on a standard formula for algorithmic models
  - PM = A ´ Size^{B} ´ M where
  - M = PERS ´ RCPX ´ RUSE ´ PDIF ´ PREX ´ FCIL ´ SCED;
  - A = 2.94 in initial calibration, Size in KLOC, B varies from 1.1 to 1.24 depending on novelty of the project, development flexibility, risk management approaches and the process maturity.

Multipliers

- Multipliers reflect the capability of the developers, the non-functional requirements, the familiarity with the development platform, etc.
  - RCPX - product reliability and complexity;
  - RUSE - the reuse required;
  - PDIF - platform difficulty;
  - PREX - personnel experience;
  - PERS - personnel capability;
  - SCED - required schedule;
  - FCIL - the team support facilities.

The reuse model

- Takes into account black-box code that is reused without change and code that has to be adapted to integrate it with new code.
- There are two versions:
  - Black-box reuse where code is not modified. An effort estimate (PM) is computed.
  - White-box reuse where code is modified. A size estimate equivalent to the number of lines of new source code is computed. This then adjusts the size estimate for new code.

Reuse model estimates 1

- For generated code:
  - PM = (ASLOC * AT/100)/ATPROD
  - ASLOC is the number of lines of generated code
  - AT is the percentage of code automatically generated.
  - ATPROD is the productivity of engineers in integrating this code.

Reuse model estimates 2

- When code has to be understood and integrated:
  - ESLOC = ASLOC * (1-AT/100) * AAM.
  - ASLOC and AT as before.
  - AAM is the adaptation adjustment multiplier computed from the costs of changing the reused code, the costs of understanding how to integrate the code and the costs of reuse decision making.

Post-architecture level

- Uses the same formula as the early design model but with 17 rather than 7 associated multipliers.
- The code size is estimated as:
    - Number of lines of new code to be developed;
    - Estimate of equivalent number of lines of new code computed using the reuse model;
    - An estimate of the number of lines of code that have to be modified according to requirements changes.

The exponent term

- This depends on 5 scale factors (see next slide). Their sum/100 is added to 1.01
- A company takes on a project in a new domain. The client has not defined the process to be used and has not allowed time for risk analysis. The company has a CMM level 2 rating.
  - Precedenteness - new project (4)
  - Development flexibility - no client involvement - Very high (1)
  - Architecture/risk resolution - No risk analysis - V. Low .(5)
  - Team cohesion - new team - nominal (3)
  - Process maturity - some control - nominal (3)
- Scale factor is therefore 1.17.

Scale factors used in the exponent computation in the post-architecture model

Scale factor	Explanation
Precedentedness	Reflects the previous experience of the organization with this type of project. Very low means no previous experience; extra-high means that the organization is completely familiar with this application domain.
Development flexibility	Reflects the degree of flexibility in the development process. Very low means a prescribed process is used; extra-high means that the client sets only general goals.
Architecture/risk resolution	Reflects the extent of risk analysis carried out. Very low means little analysis; extra-high means a complete and thorough risk analysis.
Team cohesion	Reflects how well the development team knows each other and work together. Very low means very difficult interactions; extra-high means an integrated and effective team with no communication problems.
Process maturity	Reflects the process maturity of the organization. The computation of this value depends on the CMM Maturity Questionnaire, but an estimate can be achieved by subtracting the CMM process maturity level from 5.

Multipliers

- Product attributes
  - Concerned with required characteristics of the software product being developed.
- Computer attributes
  - Constraints imposed on the software by the hardware platform.
- Personnel attributes
  - Multipliers that take the experience and capabilities of the people working on the project into account.
- Project attributes
  - Concerned with the particular characteristics of the software development project.

The effect of cost drivers on effort estimates

Exponent value	**1.17**
System size (including factors for reuse and requirements volatility)	128,000 DSI
Initial COCOMO estimate without cost drivers	**730 person-months**
Reliability	Very high, multiplier = 1.39
Complexity	Very high, multiplier = 1.3
Memory constraint	High, multiplier = 1.21
Tool use	Low, multiplier = 1.12
Schedule	Accelerated, multiplier = 1.29
Adjusted COCOMO estimate	**2,306 person-months**

The effect of cost drivers on effort estimates

Exponent value	**1.17**
Reliability	Very low, multiplier = 0.75
Complexity	Very low, multiplier = 0.75
Memory constraint	None, multiplier = 1
Tool use	Very high, multiplier = 0.72
Schedule	Normal, multiplier = 1
Adjusted COCOMO estimate	**295 person-months**

Project duration and staffing

- As well as effort estimation, managers must estimate the calendar time required to complete a project and when staff will be required.
- Calendar time can be estimated using a COCOMO 2 formula
  - $TDEV = 3 ´ (PM)^{(0.33+0.2*(B-1.01))}$
  - 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.
- The time required is independent of the number of people working on the project.

Staffing requirements

- Staff required can’t be computed by diving 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.

Key points

- Estimation techniques for software may be experience-based, where managers judge the effort required, or algorithmic, where the effort required is computed from other estimated project parameters.
- The COCOMO II costing model is an algorithmic cost model that uses project, product, hardware and personnel attributes as well as product size and complexity attributes to derive a cost estimate.