

CMPS 411

Software Project Management



*Please read
Chapter 24, 27
& 28*

Dr. Abdelkarim Erradi

Dept. of Computer Science & Engineering

QU

Outline

- Introduction to Project Management
- Work Breakdown Structures
- Dependency Graph
- Estimating times for activities
- Critical path analysis
- Project Schedule
- Risk Management

Introduction to Project Management

Definition

- A **project** is a coordinated group of interdependent tasks that are completed by people using **resources** and **processes**.
 - Tasks that must be **planned, scheduled, budgeted, staffed, and coordinated**
- Projects have **definite starting and ending dates** as well as success criteria
 - **Progressively elaborated**
- **Goal** = complete the project on time, within budget, and according to specifications (while dealing with complexity and change)

Project components

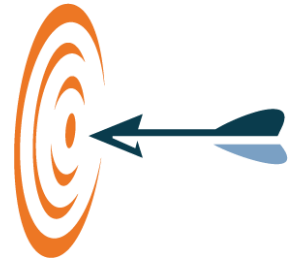
- There are four interrelated components of any project:
 - **Scope:** deliverables, features and functions, acceptance criteria, constraints, and risks.
 - **Schedule:** includes the beginning and ending times and duration for all project tasks
 - **Resources:** all assets needed to complete the project
 - **Leadership:** controls and monitors the project execution

People skills of Project Manager

- Project managers perform both **process** and **people** functions:
 - Teambuilding: **Matching people to tasks**
 - Leadership: Create a sense of vision and excitement.
 - Motivation
 - Communication: keep stakeholders up to date throughout the project
 - Time management
 - Change management
 - Dealing with adversity: remove road blocks

Project Phases

- Projects have five distinct phases:
 - Initiation: develop a **Project charter**
 - **Planning**: Work Breakdown Structure and Schedule
 - Execution: execute the project tasks
 - Monitoring/control of the **scope, schedule, costs, quality, and risk.**
 - Closing: reflect and document the lessons learnt



Project Initiation

- The outcomes of the project initiation phase of a project are:
 - project description, feasibility analysis report, concept document, **project charter** with scope, stakeholder register, and the project kickoff meeting.
 - **Project charter** contains a project overview, assumptions, scope, milestones, deliverables, resources, funding, and project agreement.

Project Planning

Project Planning Activities

- Identify activities
- **Estimation:** Effort, cost, resource, and project duration
- Project **scheduling:** determine the sequence of activities and assign resources
- **Risk** handling:
 - identification, analysis, and mitigation
- **Miscellaneous plans:**
 - quality assurance plan, configuration management plan, etc.

What is the problem?

- Your boss: “How long will the project take?”
- In reality, you don’t have the slightest clue how long it will take, because **you don’t know the work to be done.**
- Solution: Use divide and conquer
 - To give a good answer you have to break the work down into activities for which you can get good timing estimates
 - From these estimates you compute the estimated project duration












Activities to obtain good time estimates

- Identify the work that needs to be done
 - Work breakdown structure (WBS)
- Identify the dependency between work units
 - Dependency Graph
- Estimate the duration of the work to be done
 - Schedule

Project Plan and Schedule

- Work Breakdown Structure (WBS)
 - Sub-divide project deliverables into smaller and smaller activities
 - Hierarchical decomposition of the project into activities and tasks
- Dependencies between tasks
 - Dependency graphs showing temporal dependencies of the activities
 - Add Time Estimates
 - Determine the **critical path**
- Visualize the activities on a time scale: **Gantt Chart**

Software Project Management Plan (IEEE Std 1058)

-  0. Front Matter
-  1. Introduction
-  2. Project Organization
-  3. Managerial Process
-  4. Technical Process
-  **5. Work Elements, Schedule, Budget**
 -  5.1 Work Breakdown Structure (WBS)
 -  5.2 Dependencies between tasks
 -  5.3 Resource Requirements
 -  5.4 Budget
 -  5.5 Schedule
- Optional Inclusions

Let's Build a House

- What are the activities that are needed to build a house?

1) Identify the work to be done: Work Breakdown Structure

- Surveying
- Excavation
- Request Permits
- Buy Material
- Lay foundation
- Build Outside Wall
- Install Exterior Plumbing
- Install Exterior Electrical
- Install Interior Plumbing
- Install Interior Electrical
- Install Wallboard
- Paint Interior
- Install Interior Doors
- Install Floor
- Install Roof
- Install Exterior Doors
- Paint Exterior
- Install Exterior Siding

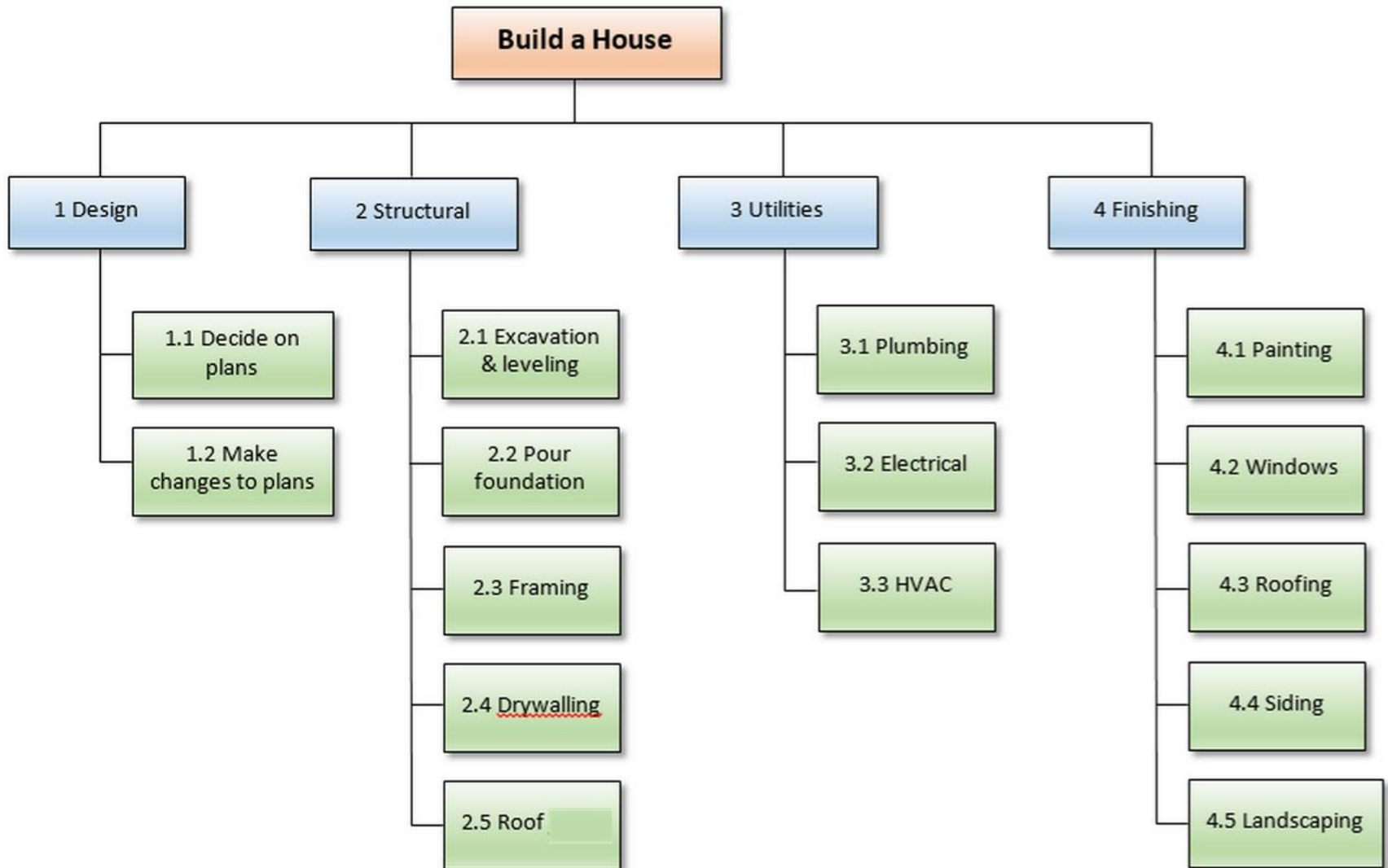
Finding these activities is a brainstorming activity. Similar activities are identified during requirements engineering and analysis (use case modeling)

2) Hierarchically organize the activities

- Building the house consists of
 - Prepare the building site
 - Building the Exterior
 - Building the Interior
- Preparing the building site consists of
 - Surveying
 - Excavation
 - Buying of material
 - Laying of the foundation
 - Requesting permits

Finding this organization involves categorization and refinement. Good after (not during) brainstorming

WBS for building a house

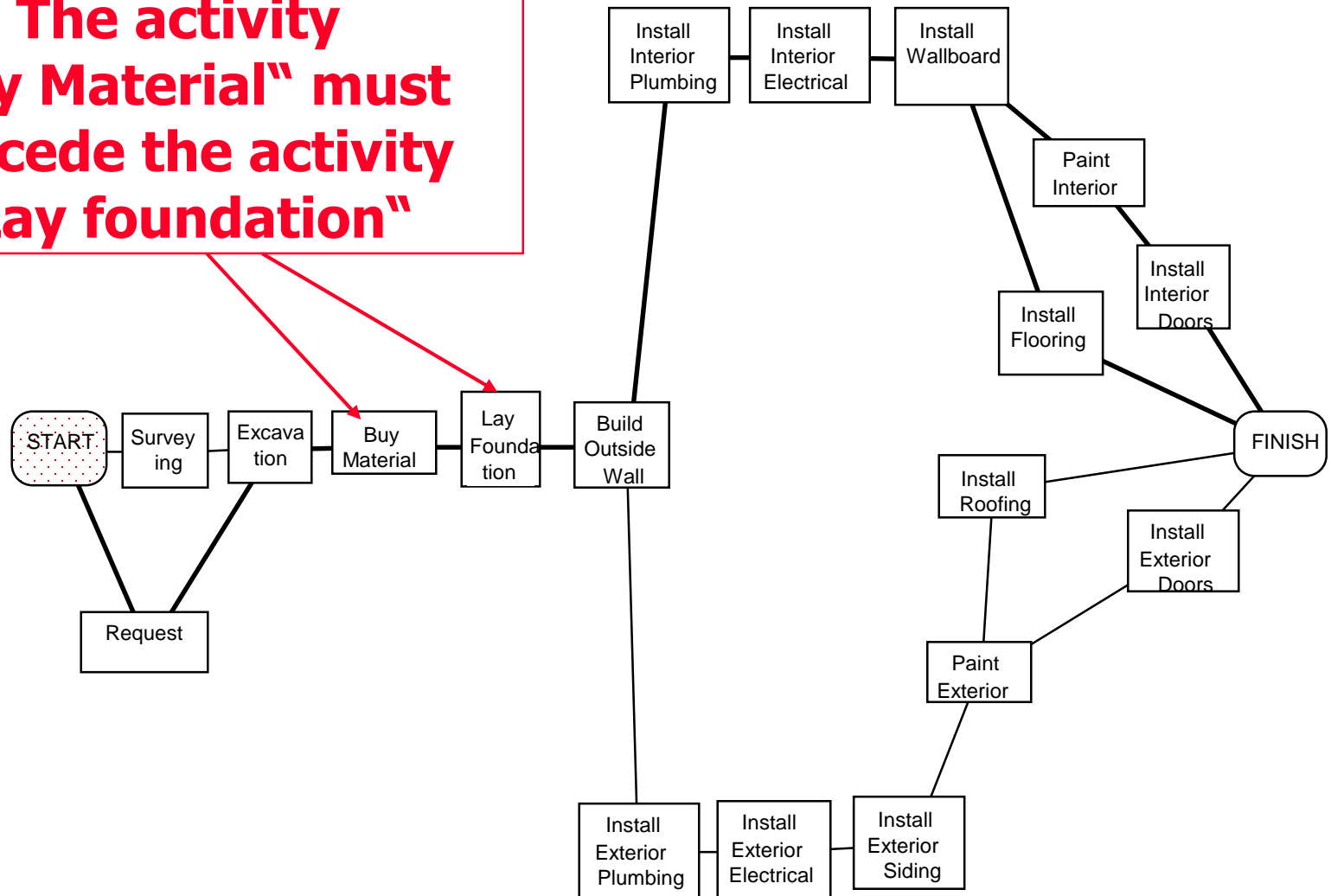


3) Identify dependencies between tasks

- **The work breakdown structure does not show any dependence among the activities/tasks**
 - Can we excavate before getting the permit?
 - How much time does the whole project need if I know the individual times?
 - What can be done in parallel?
 - Are there any critical activities, that can slow down the project significantly?
- **Dependencies like these are shown in the dependency graph (also called Network Graph)**
 - **Nodes** are activities
 - **Lines** represent temporal dependencies

Building a House (Dependency Graph)

**The activity
"Buy Material" must
Precede the activity
"Lay foundation"**



4) Map tasks onto time

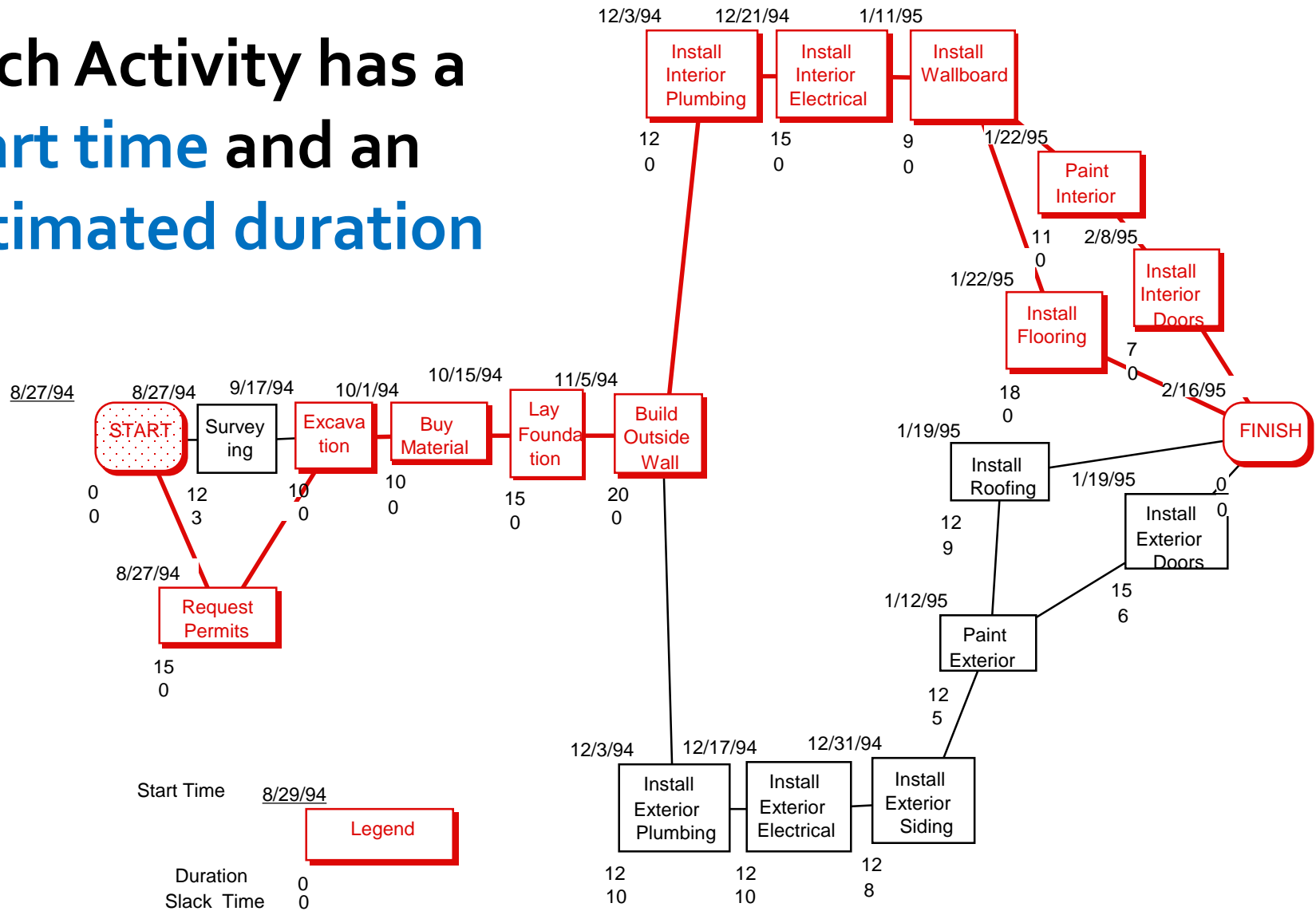
- Estimate starting times and durations for each of the activities in the dependency graph
- Compute the longest path through the graph:
This is **the estimated duration of your project**

How do we get good estimate times?

- Estimation activity durations is crucial for setting up a plan.
- We will discuss methods and heuristics on how to do it and how to establish a project schedule.

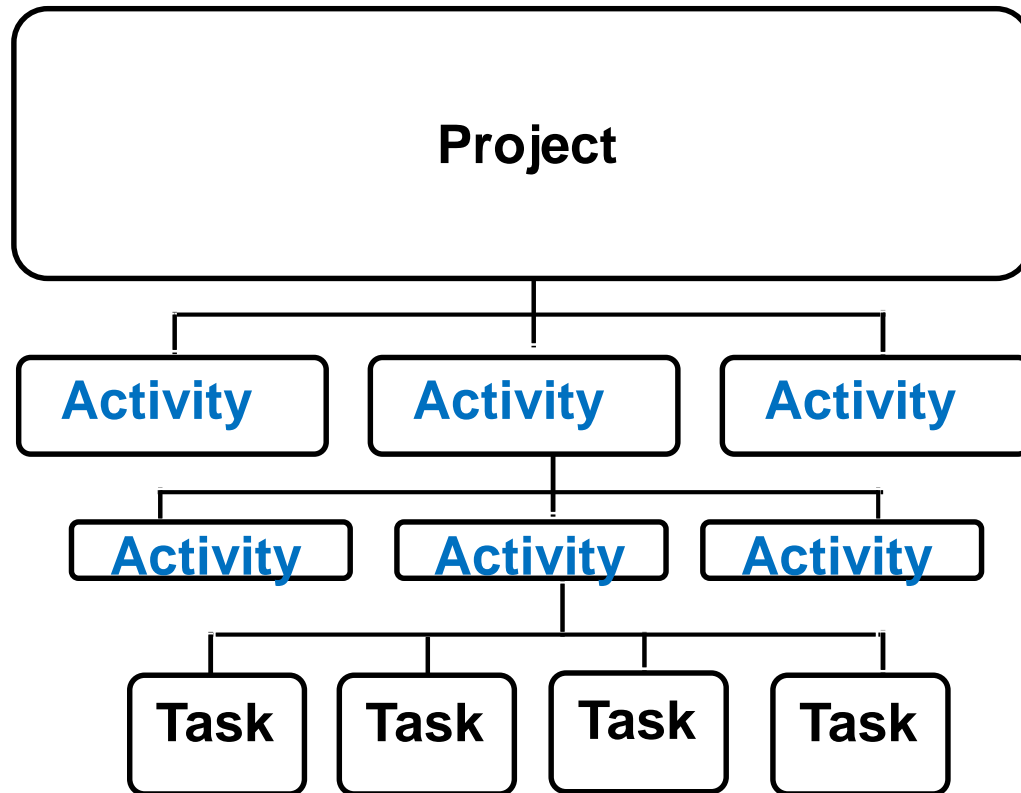
Building a House Schedule

Each Activity has a
start time and an
estimated duration



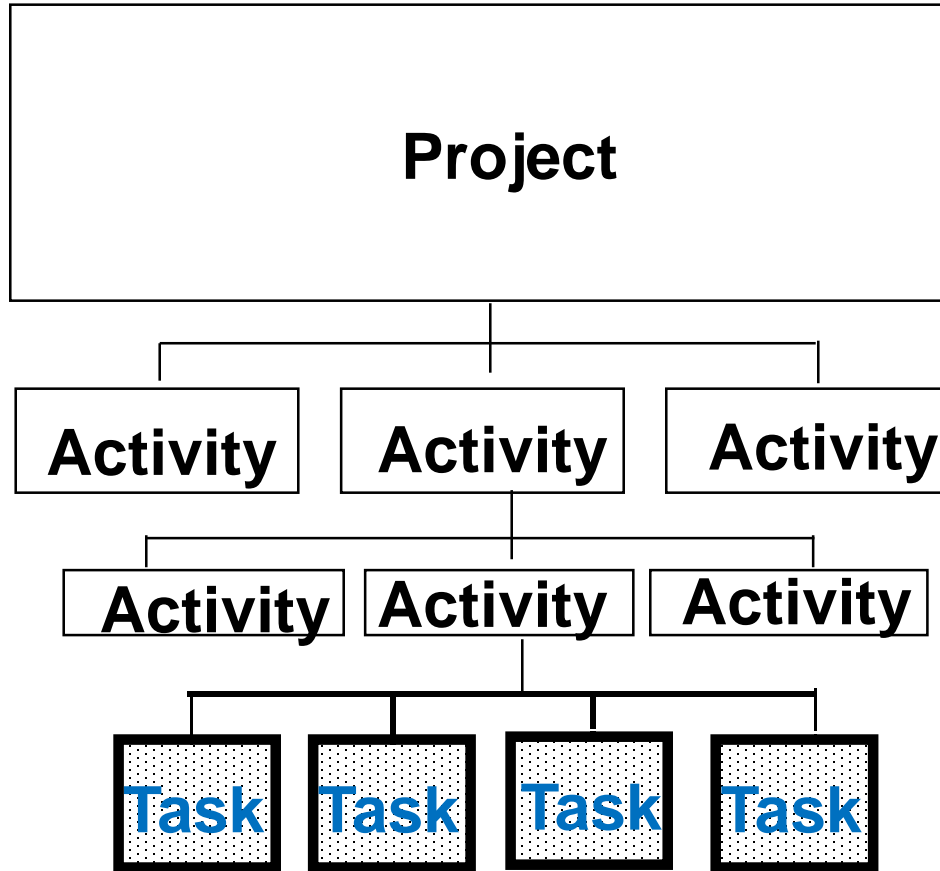
Activities and Tasks

A Project has a duration and consists of activities and tasks



- A major unit of work that **requires performing tasks to produce a deliverable.**
- Consists of smaller activities or tasks
- Has an **expected duration.**
- Consumes **costs and resources.**
- Named in **verb-noun format.** (Update Requirements Documentation, configure hardware, install software...etc.)

Tasks



- Smallest unit of work subject to management
- Small enough for adequate planning and tracking (e.g., Unit test class Foo)
- Large enough to avoid micro management
- Heuristics for Duration: be done by a person within one week or two weeks

Ways to Create WBS

- Two major ways:
 - **Activity-oriented decomposition** (“Functional decomposition”)
 - Write the book
 - Get it reviewed
 - Do the suggested changes
 - Get it published
 - **Result-oriented** (“Architecture based decomposition”)
 - Chapter 1
 - Chapter 2
 - Chapter 3

WBS - Functional decomposition

- **Functional decomposition**
 1. Software system
 - 1.1 Functionality 1 (or Use case 1)
 - 1.1 Functionality 2 (or Use case 2)
- Decompose the use cases (work packages) into tasks that need to be completed to deliver the use case.

WBS - Architecture based decomposition

- Architecture based decomposition
 1. Software system
 - 1.2 Subsystem 1
 - 1.2.1 Component 1
 - 1.2.2 Component 2
- Decompose the components (work packages) into tasks that need to be completed to produce the particular component.
- The problem is that to make this kind of WBS you need to have an architecture design completed or at least well advanced which is not always the case at the beginning of the project.
- Moreover this kind of WBS may not be easily understandable for the customer and management who have functional rather than technical understanding of the system.

Heuristics for developing high quality WBS

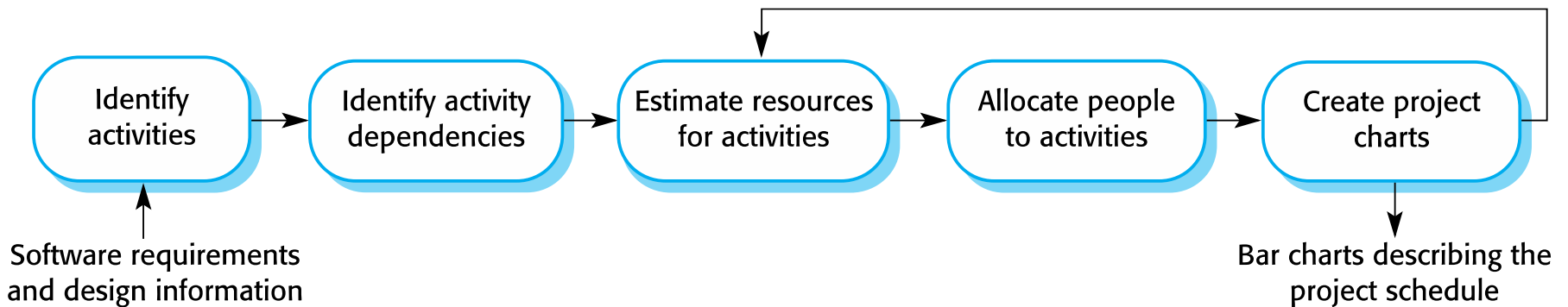
- Involve the people who will be doing the work in the development of the WBS
 - In particular involve the developers
- Review and include information from work breakdown structures that were developed for similar projects
 - Use a project template if possible
- Use more than one WBS approach
 - Do project component and functional approach simultaneously
 - This allows you often to identify overlooked activities

Schedule

Four Steps:

1. WBS
2. Determine the Dependency Diagram
4. Add Time Estimates
5. Visualize the activities on a time scale: Gantt Chart

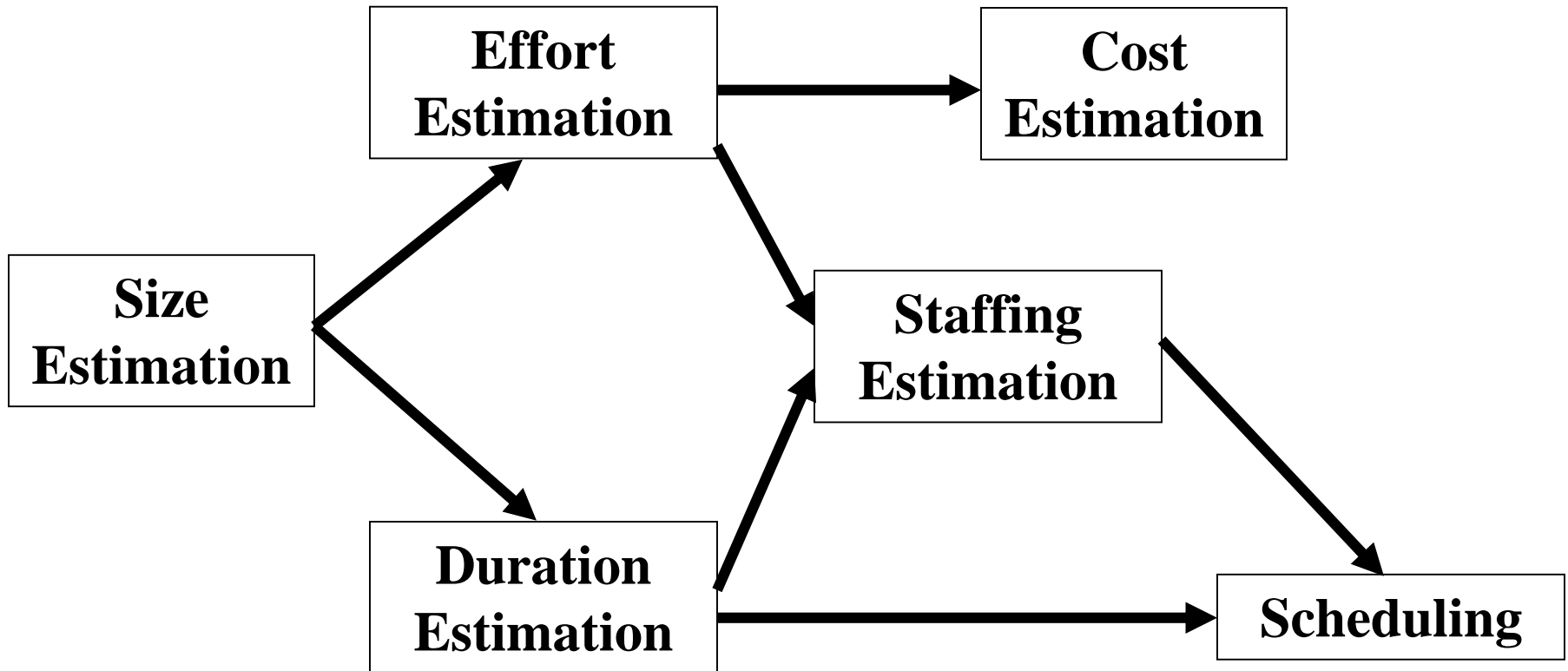
Project scheduling process



Software Cost Estimation

- Determine size of the product
- From the size estimate determine the effort needed
- From the effort estimate determine project duration and cost.

Software Cost Estimation



Why Dependency Diagrams?

- Dependency Diagrams are a formal notation to help in the construction and analysis of *complex schedules*
- Compute the project duration
- Determine the critical path i.e., activities that are critical to ensure a timely delivery
- Analyse the diagrams
 - to find ways to shorten the project duration
 - To find ways to do activities in parallel
- 2 techniques are used
 - Forward pass (determine critical paths)
 - Backward pass (determine slack time)

Definitions: Critical Path and Slack Time

- **Critical path:**
 - A sequence of activities that take the longest time to complete
 - The length of the critical path(s) defines how long the project will take to complete.
- **Noncritical path:**
 - A sequence of activities that you can delay and still finish the project in the shortest time possible.
- **Slack time:**
 - The maximum amount of time that you can delay an activity and still finish your project in the shortest time possible.

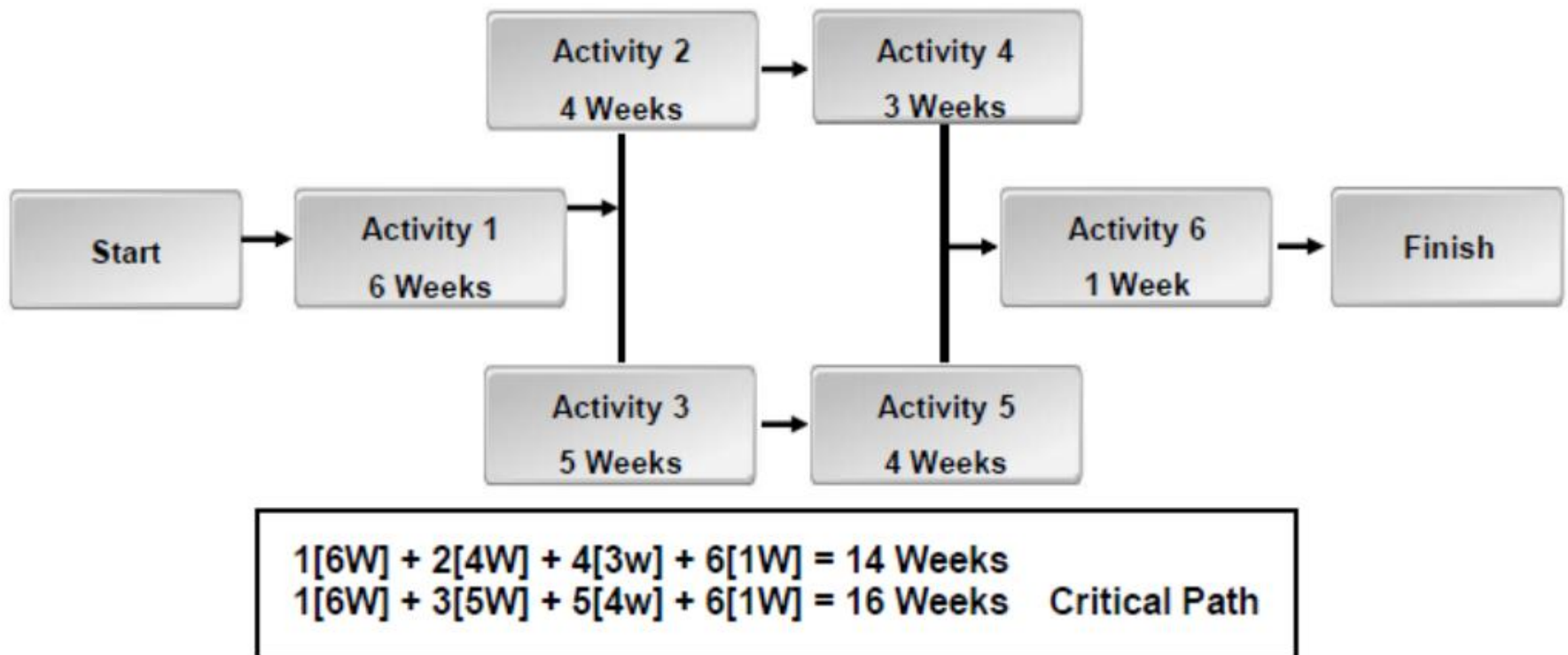
Standard Schedule Diagramming Notations

Relationship	Description
ES	Early start. The earliest time an activity can start.
EF	Early finish. The earliest time an activity can finish.
LF	Late finish. The latest time an activity can finish.
LS	Late start. The latest time an activity can start.
DU	Duration. The number of work periods required for completion of an activity.

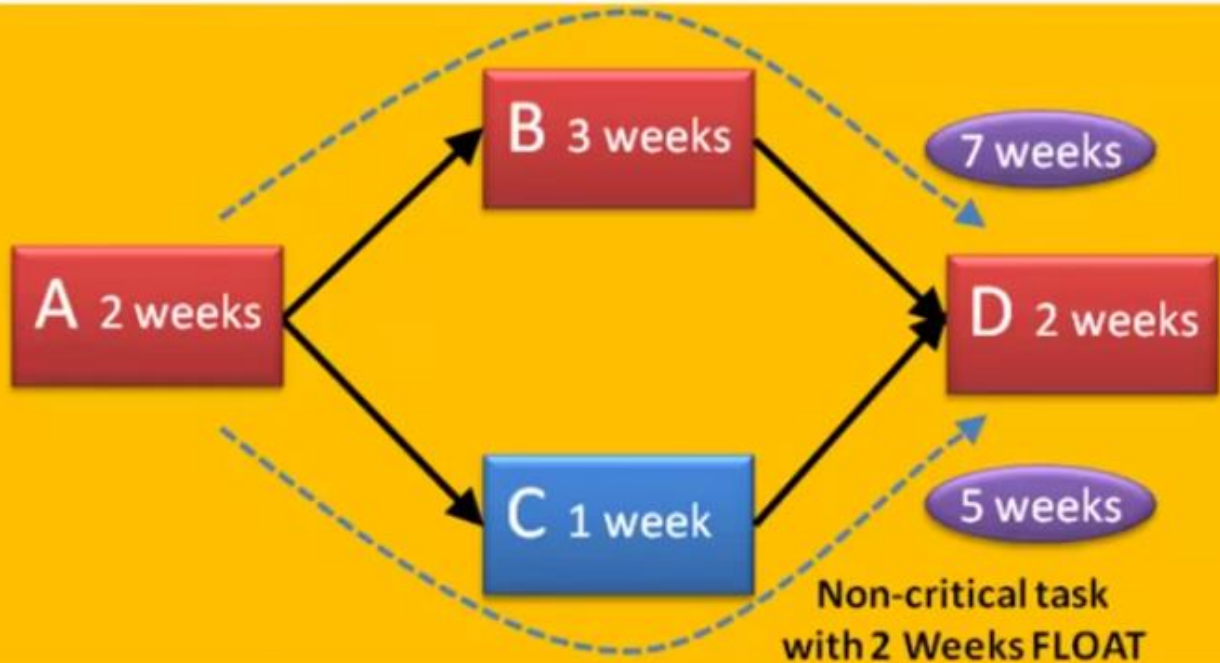


Critical Path

- Critical Path = Path with the longest total duration.
 - Has no scheduling flexibility.



Critical Path Analysis

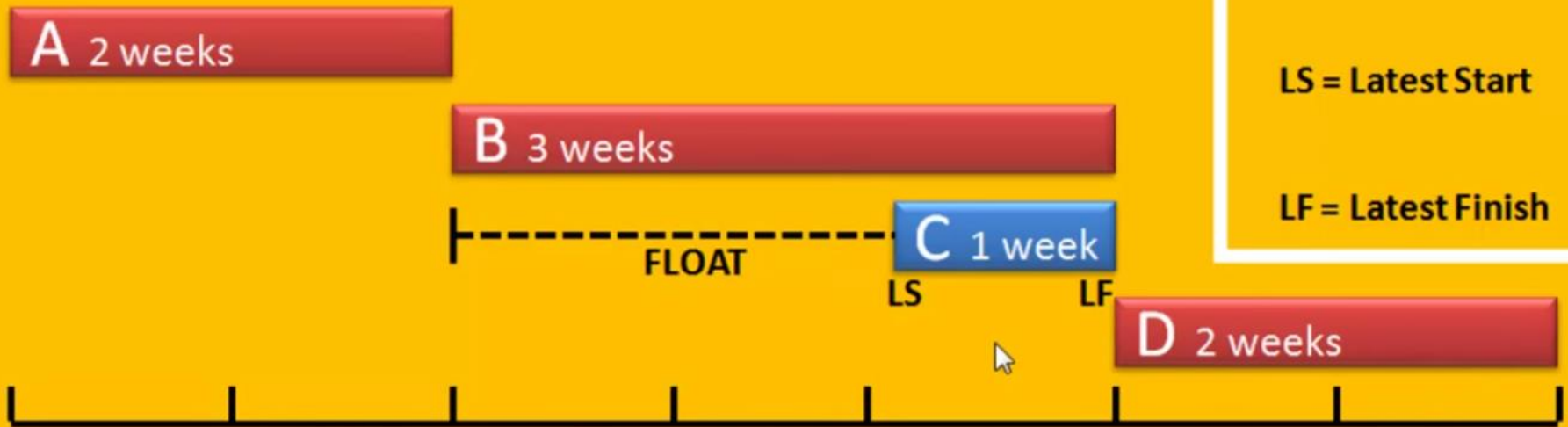


The longest path determines the EARLIEST project finish time

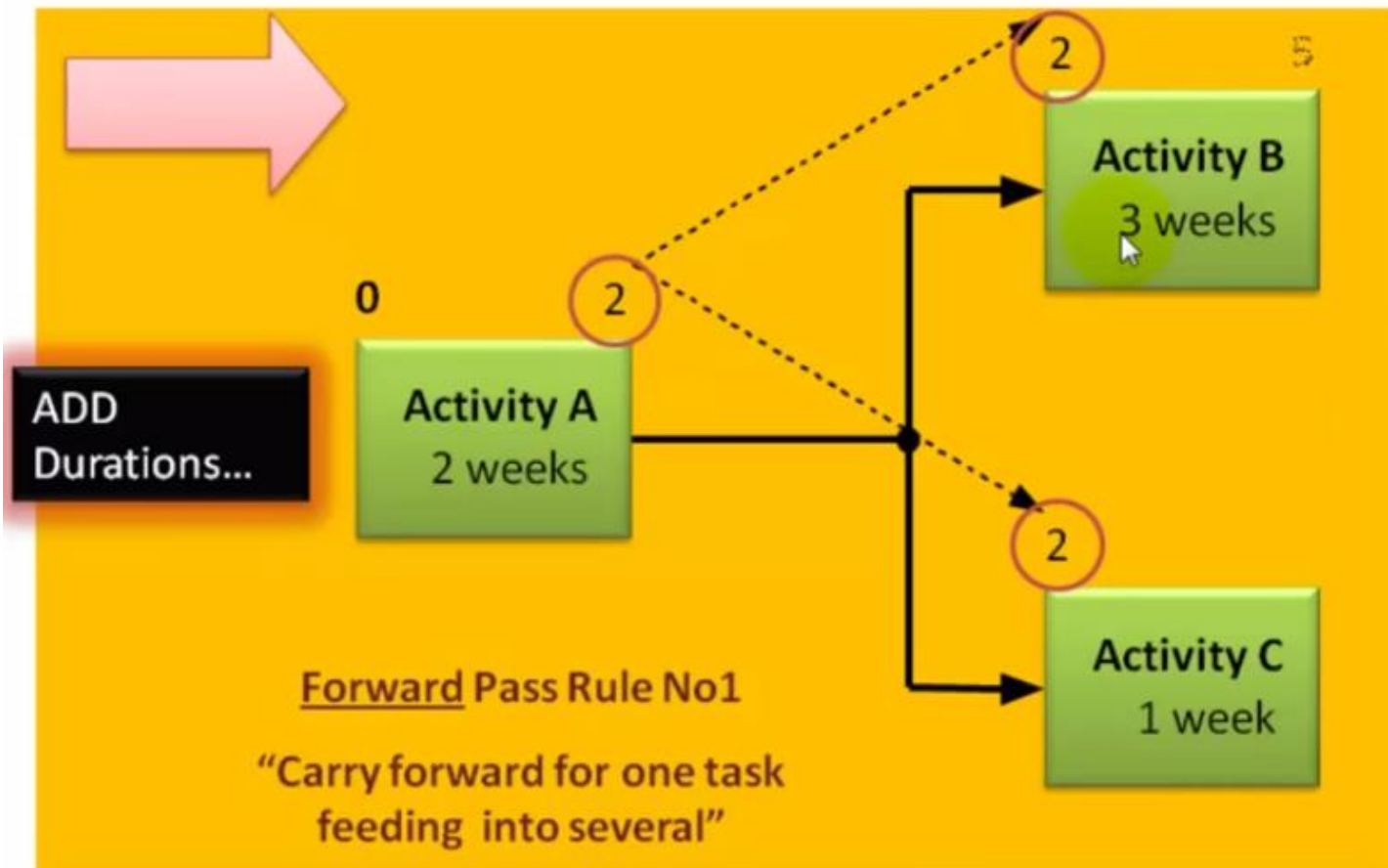
The longest path is called the (time) CRITICAL PATH

Tasks on the critical path are called critical tasks

Non-critical tasks have 'FLOAT' or 'SLACK'



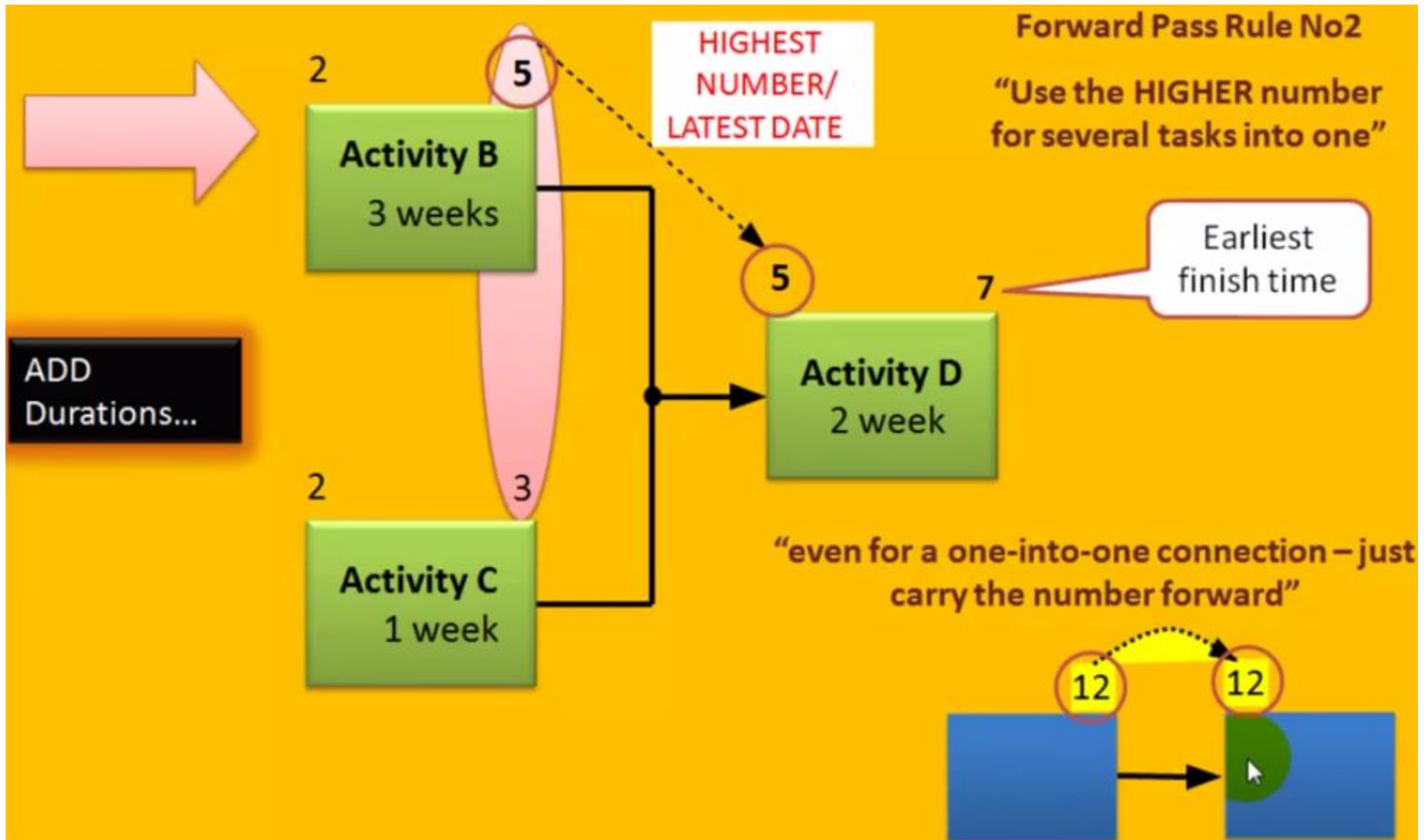
Forward Pass Calculation



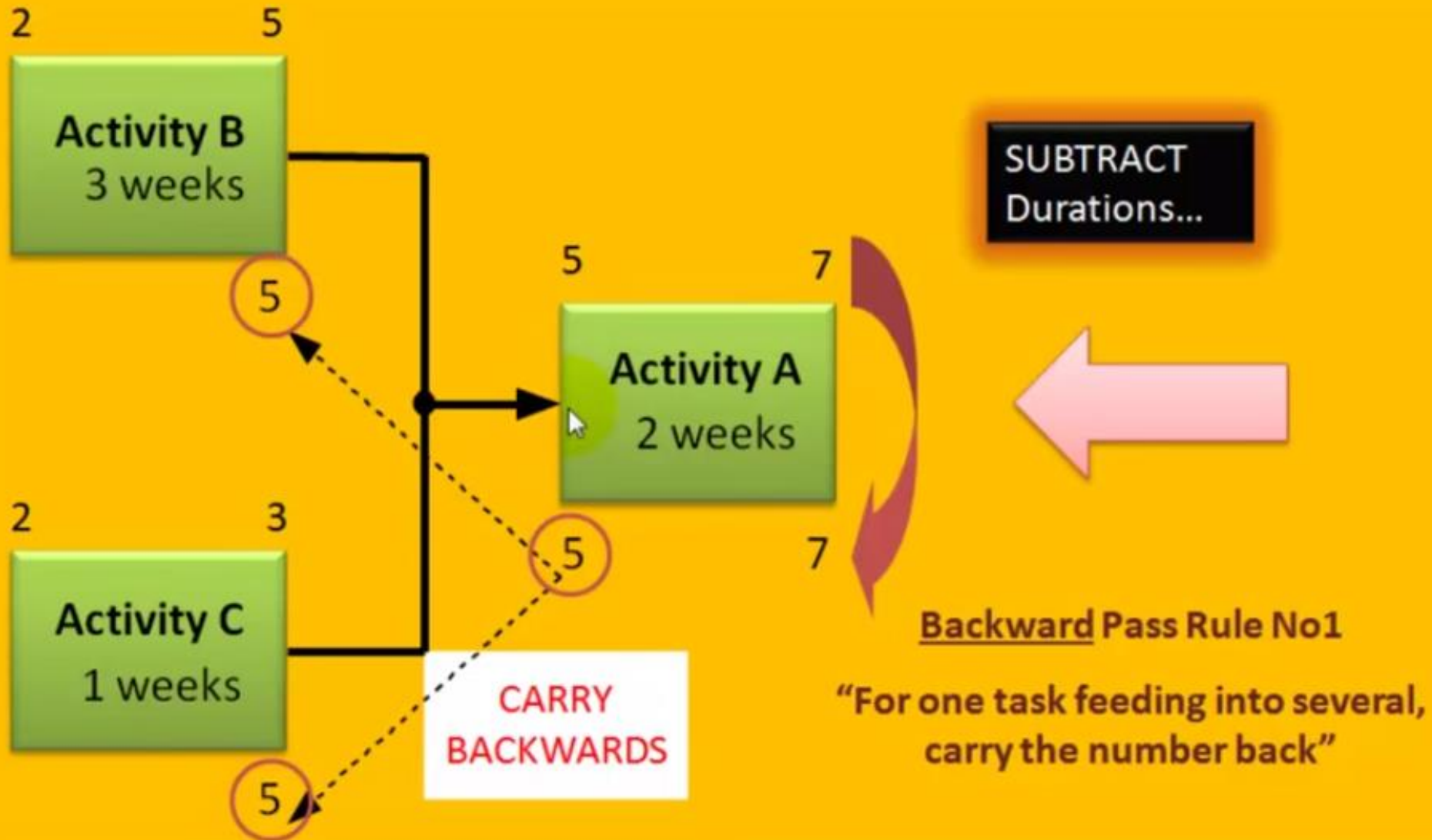
The forward pass calculates the earliest finish date ...

The backward pass calculates the critical path and the slack (float) of non-critical tasks...

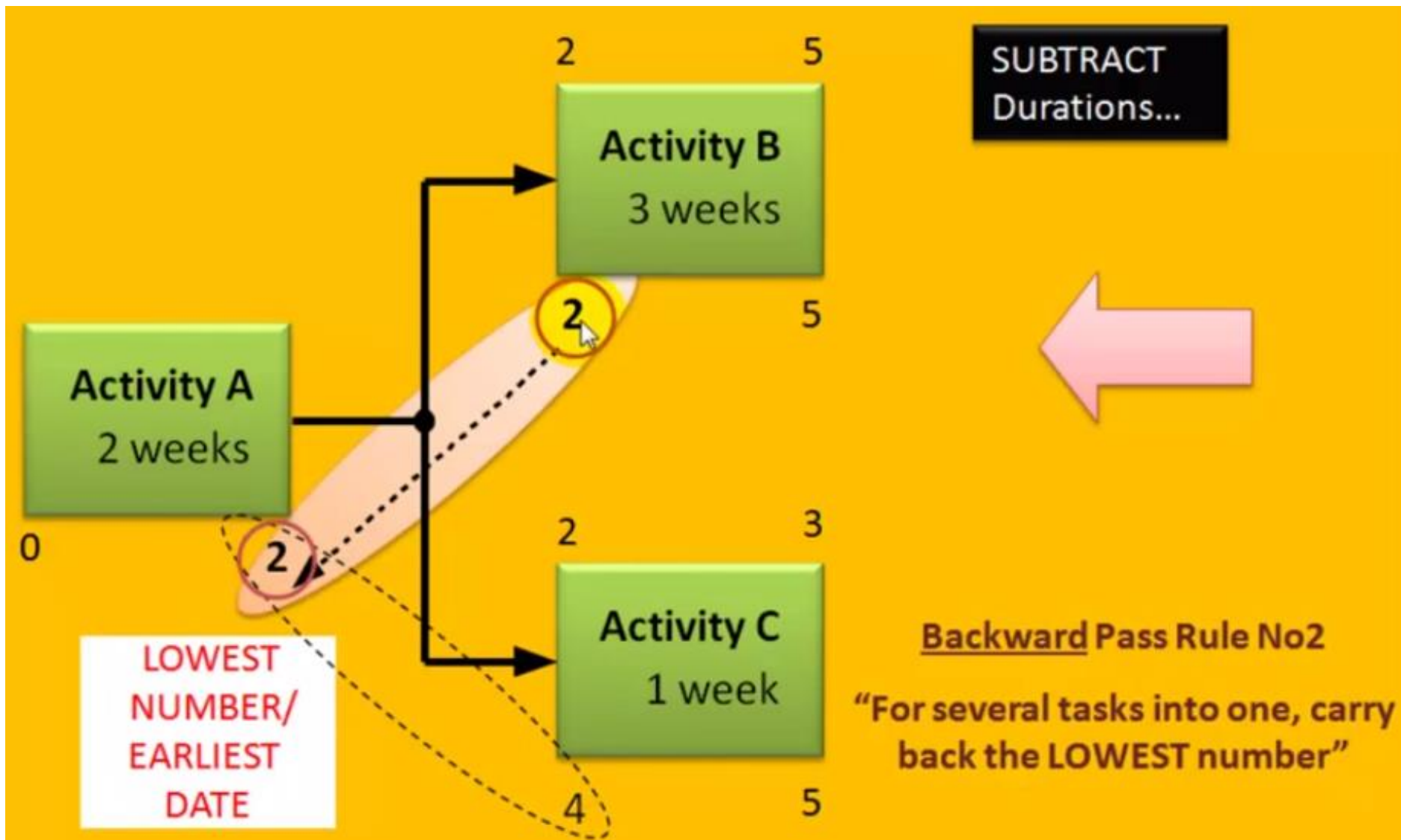
Forward Pass Calculation 2



Backward Pass Calculation



Backward Pass Calculation 2



Critical Path and Float

Duration

ES	Duration	EF
Activity Name & Float		
LS	Duration	LF

ANALYSIS

“Using EITHER the front pair OR the Back pair.....”

If difference is zero,
task is **CRITICAL**

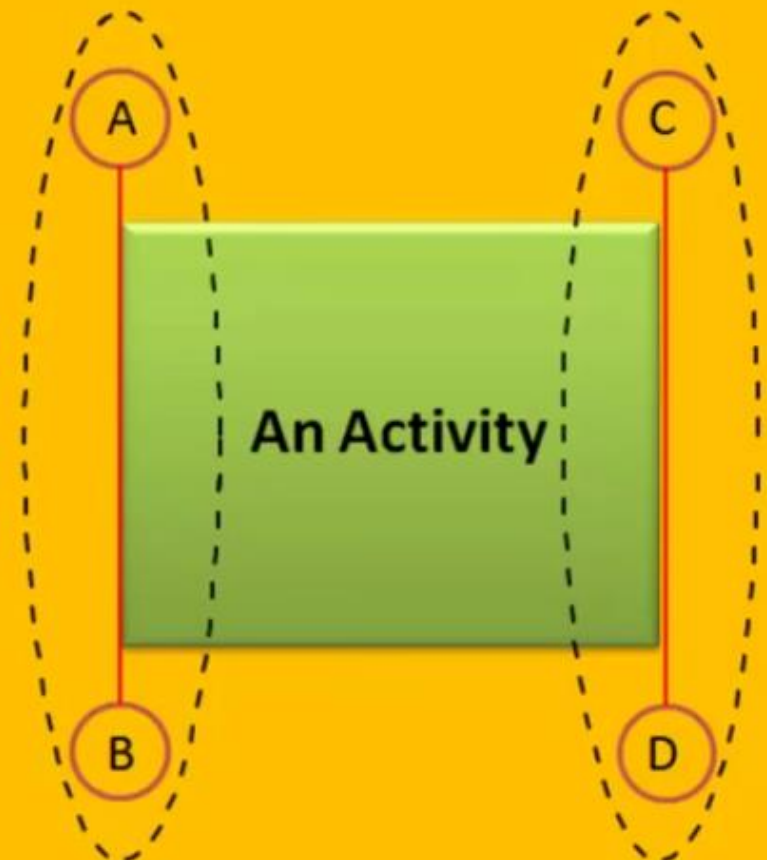
If difference is non-zero,
task is **NON-CRITICAL**
and difference = Float

ES = Early Start

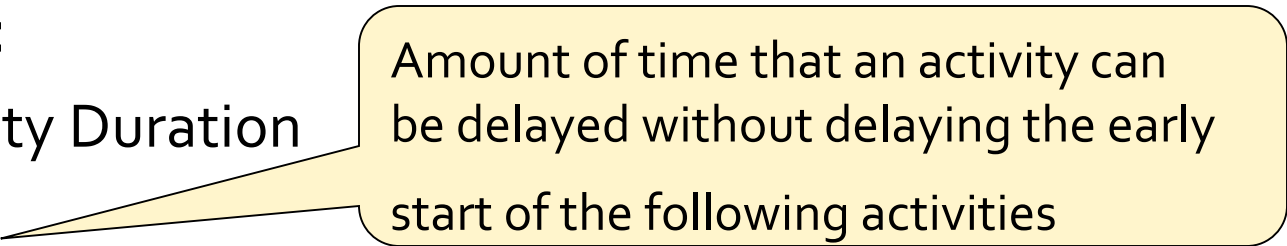
EF = Early Finish

LS = Latest Start

LF = Latest Finish

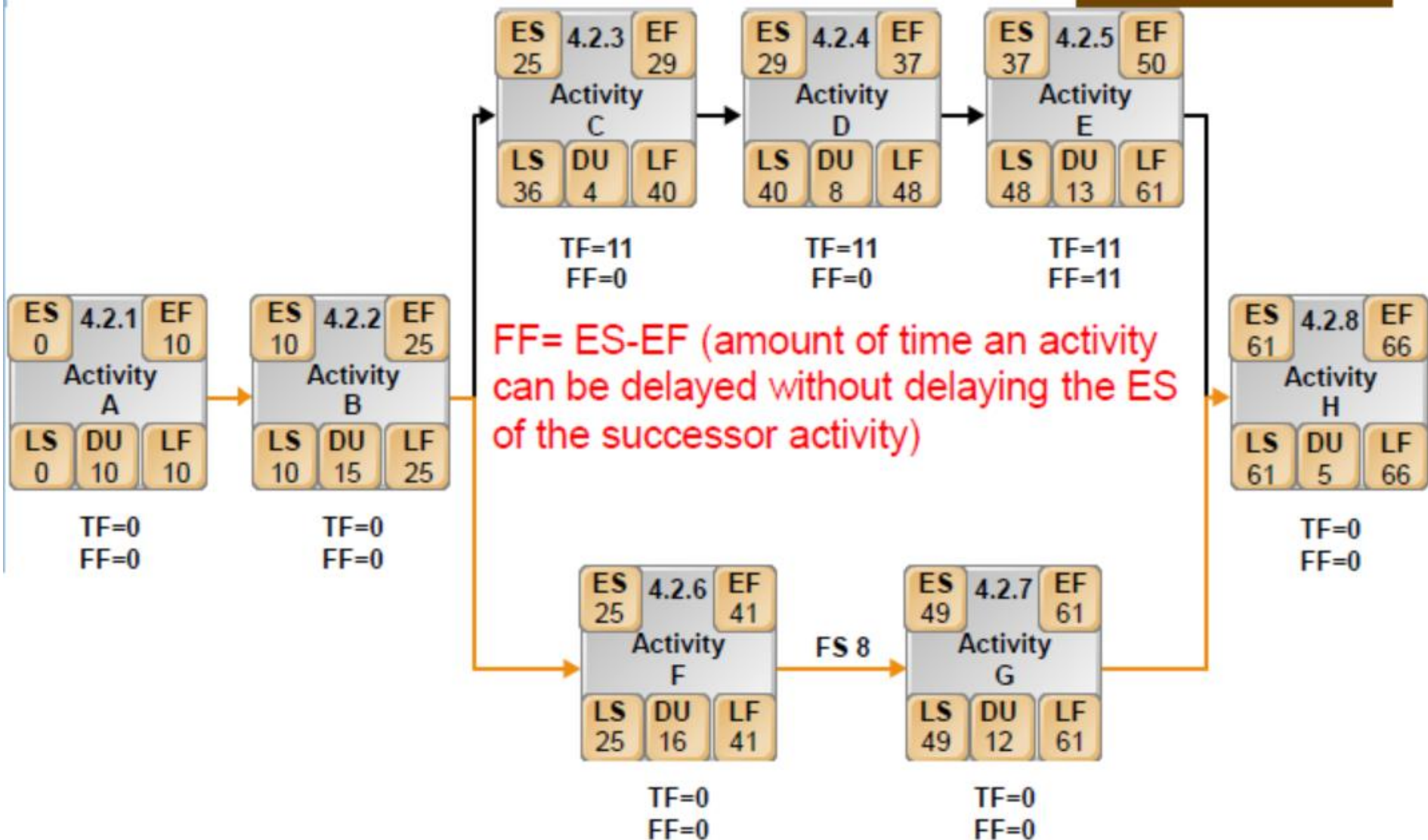


Critical Path Analysis

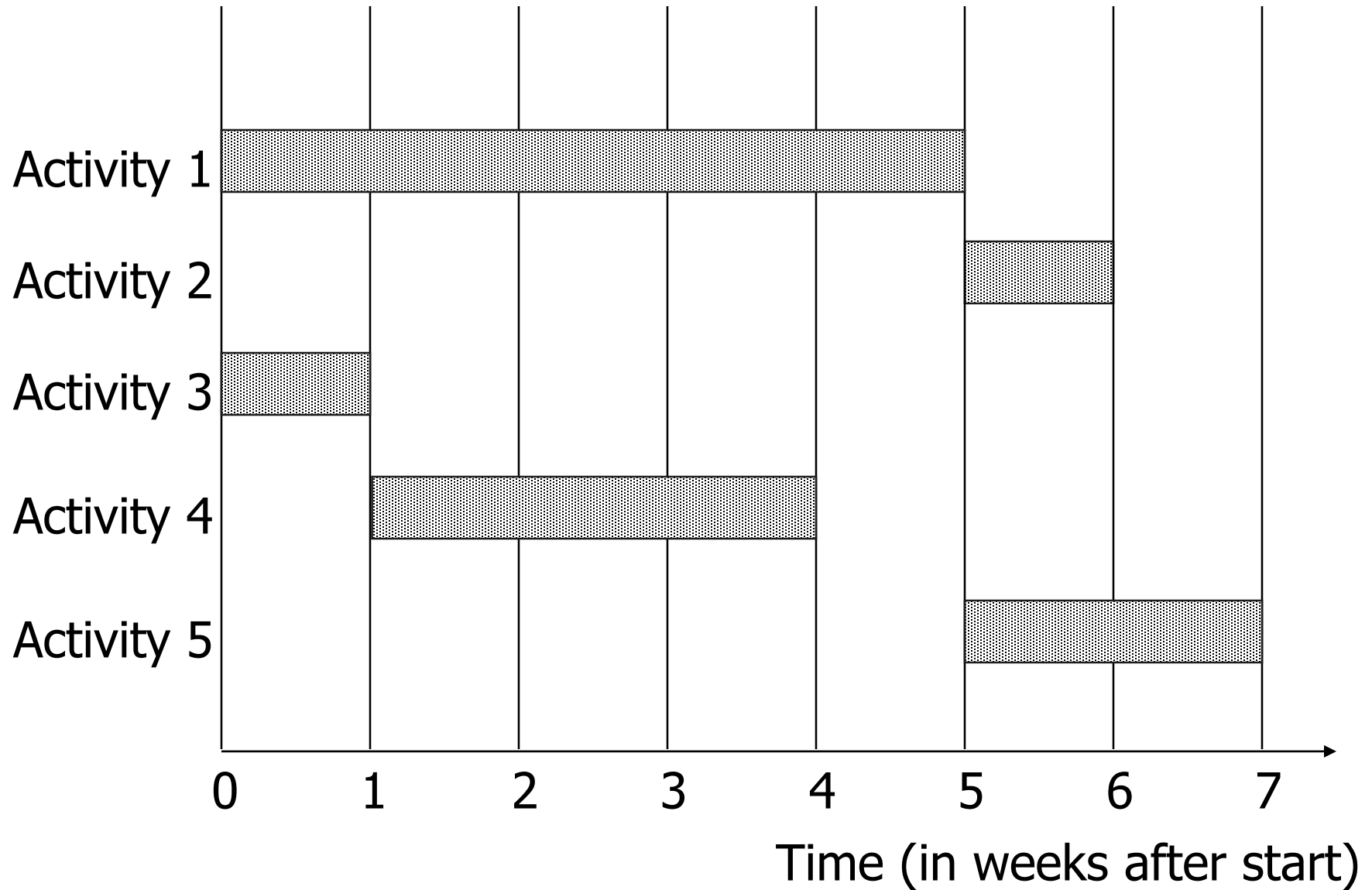
- Conduct a forward pass to determine early start (ES) and early finish (EF) times for each activity.
- Forward pass:
 - $ES = EF \text{ of Preceding activity}$
 - $EF = ES + \text{Activity Duration}$
- Perform a backward pass to determine late start (LS) and late finish (LF) times for each activity.
- Backward pass:
 - $LS = LF - \text{Activity Duration}$
- Calculate Float 

Amount of time that an activity can be delayed without delaying the early start of the following activities
- Identify the critical path as the path with the longest total duration and zero float.

Critical Path Analysis – Example 2



Gantt Chart



Schedule Compression

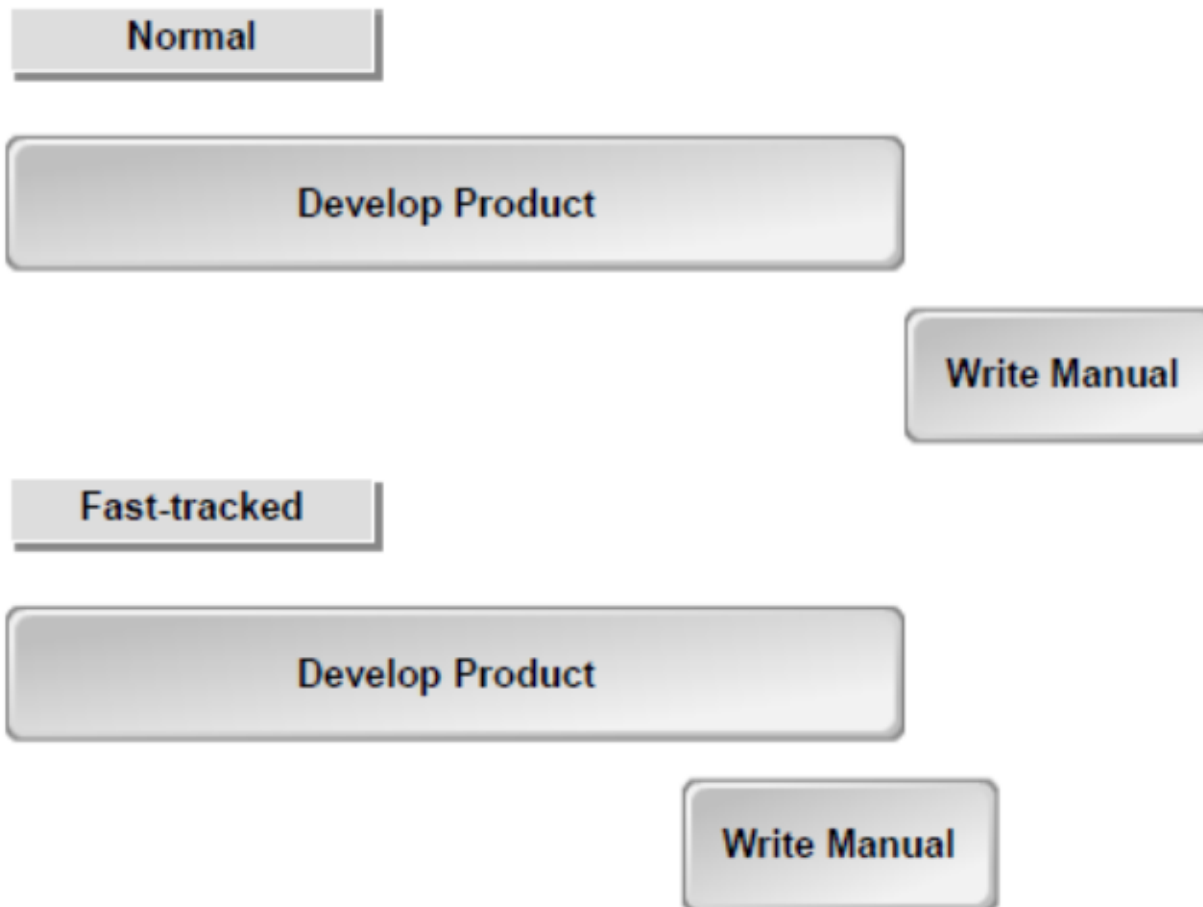
- Then try to reduce the planned project time by considering ways to shorten the aggregate time it takes to complete the critical path.
- Schedule compression shortens the project schedule without changing the project scope, to meet schedule constraints, imposed dates, or other schedule objectives.
- Schedule compression techniques include: **Crashing & Fast tracking**
- Avoid **fudge factors**, i.e., the extra amount of time you add to your best estimate for “just to be safe”

Crashing

- A specific type of project schedule compression technique performed by taking action to decrease the total project schedule duration after analyzing a number of alternatives to determine how to get the maximum schedule duration compression for the least additional cost.
- Techniques: Add resources assigned to a task, limit requirements/reduce duration, change task sequence
- Can result in increased cost

Fast-Tracking

Fast tracking = Overlapping of activities



How to reduce the planned project time

- Recheck the task estimates
 - Ask other experts to check the estimates
 - Has the development environment changed? (batch vs. interactive systems, desktop vs Web development)
- Hire more experienced personnel to perform the activities
 - Trade-off: Experts work fast, but cost more
- Consider different strategies to perform the activities
 - Consider to Buy a work product instead of building it (Trade-off: Buy-vs-build)
 - Consider external subcontractor instead of performing the work internally
- Try to find parallelizable activities on the critical path
 - Continue coding while waiting for the results of a review
- Develop an entirely new strategy to solve the problem

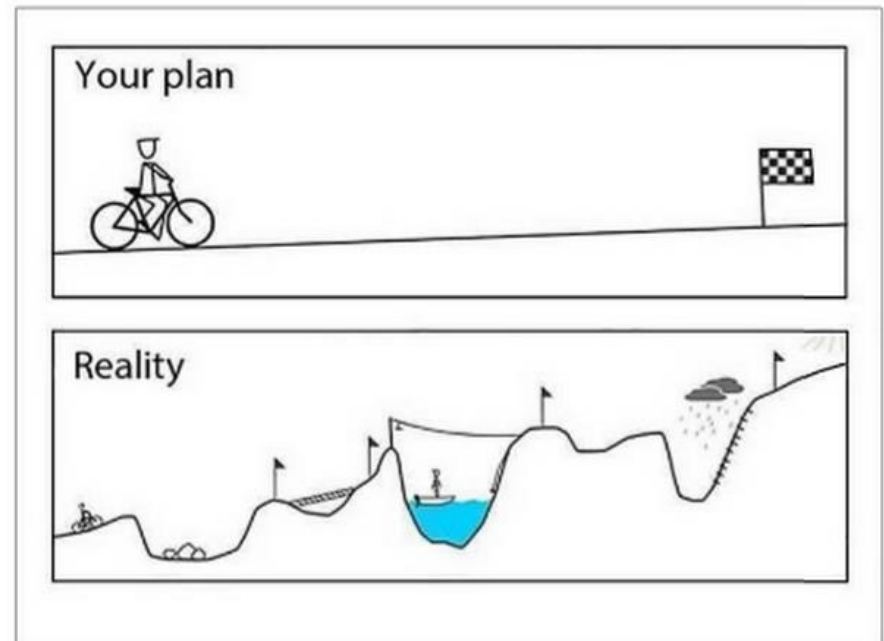
Summary

- Good scheduling is the key to successful project management
- **Work Breakdown Structure (WBS):** Set of activities to do (“use cases”)
- **Dependency Graph** (also known as **Schedule Network Diagram**): Identification of dependency relationships between activities identified in the WBS
- **Schedule:** Dependency graph decorated with time estimates for each activity
- **Gantt Chart:** Simple notation used to visualize a schedule

Risk Management

Risk Management Plan

- Risk = the possibility that things will not go as planned and that may delay the project or affect the quality of the deliverables.
- Several factors can increase the level of risk in a project:
 - inaccurate estimates
 - inexperience
 - unfamiliarity



Risk Management Plan

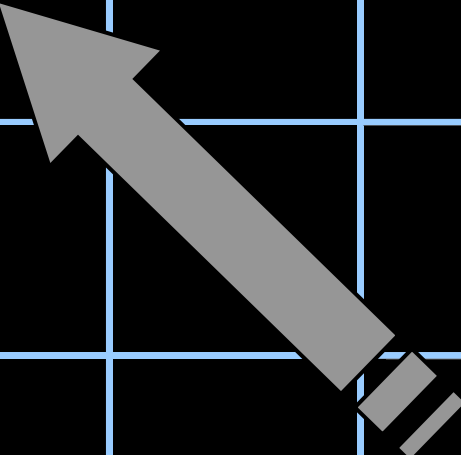
- Risk management involves:
 - identifying risks
 - assessing their potential impact
 - developing risk-mitigation plans (to eliminate/minimize the risk and its impact)
 - implementing the plans in ways that minimize the risk

Risk Map

		Likelihood of Occurrence		
		High	Medium	Low
Scale of impact	Large			
	Medium			
	Small			

Risk Map

		Likelihood of Occurrence		
		High	Medium	Low
Scale of impact	Large			
	Medium			
	Small			



Focus more the risks with high likelihood of occurring and with high impact

Risk Management Examples

- Risk: Members in key roles leave the project.
 - *Contingency Plan?*
 - Roles are assigned to somebody else. Functionality of the system is renegotiated with the client.
- Risk: The project is falling behind schedule.
 - *Contingency Plan?*
 - Extra project meetings are scheduled.
- Risk: Team 1 cannot provide functions needed by team 2.
 - *Contingency Plan?*
 - The liaisons of both teams get together to solve this problem
- Risk: The HPC computer will not be available.
 - *Contingency Plan?*
 - We will use an HPC in the cloud instead.

Risk Management Examples cont.

- Risk: The selection of the DBMS takes too much time
 - Contingency Plan?
 - The Database team uses a bridge pattern and provides a test stub to be used by the other teams for data access while the selection process goes on.
- Risk: The customer is not available for discussing and reviewing the user interface during development.
 - Contingency Plan?
 - Make the design decisions that we feel are appropriate
- Risk: No suitable wireless library can be found.
 - Contingency Plan?
 - The wireless team develops its own library