Lecture Slides for
Managing and Leading Software Projects

Chapter 5: Project Planning Techniques

developed by

Richard E. (Dick) Fairley, Ph.D.

to accompany the text

Managing and Leading Software Projects

published by Wiley, 2009

Chapter 5 Contents

- Introduction to Project Planning Techniques
- Objectives of This Chapter
- The Scope of Planning
- Rolling-Wave Planning
- Scenarios for Developing a Project Plan
- Developing the Architecture Decomposition View and the Work Breakdown Structure
- Guidelines for Designing Work Breakdown Structures
- Developing the Project Schedule
- Developing Resource Profiles
- Resource-Gantt Charts
- Estimating Project Effort, Cost, and Schedule

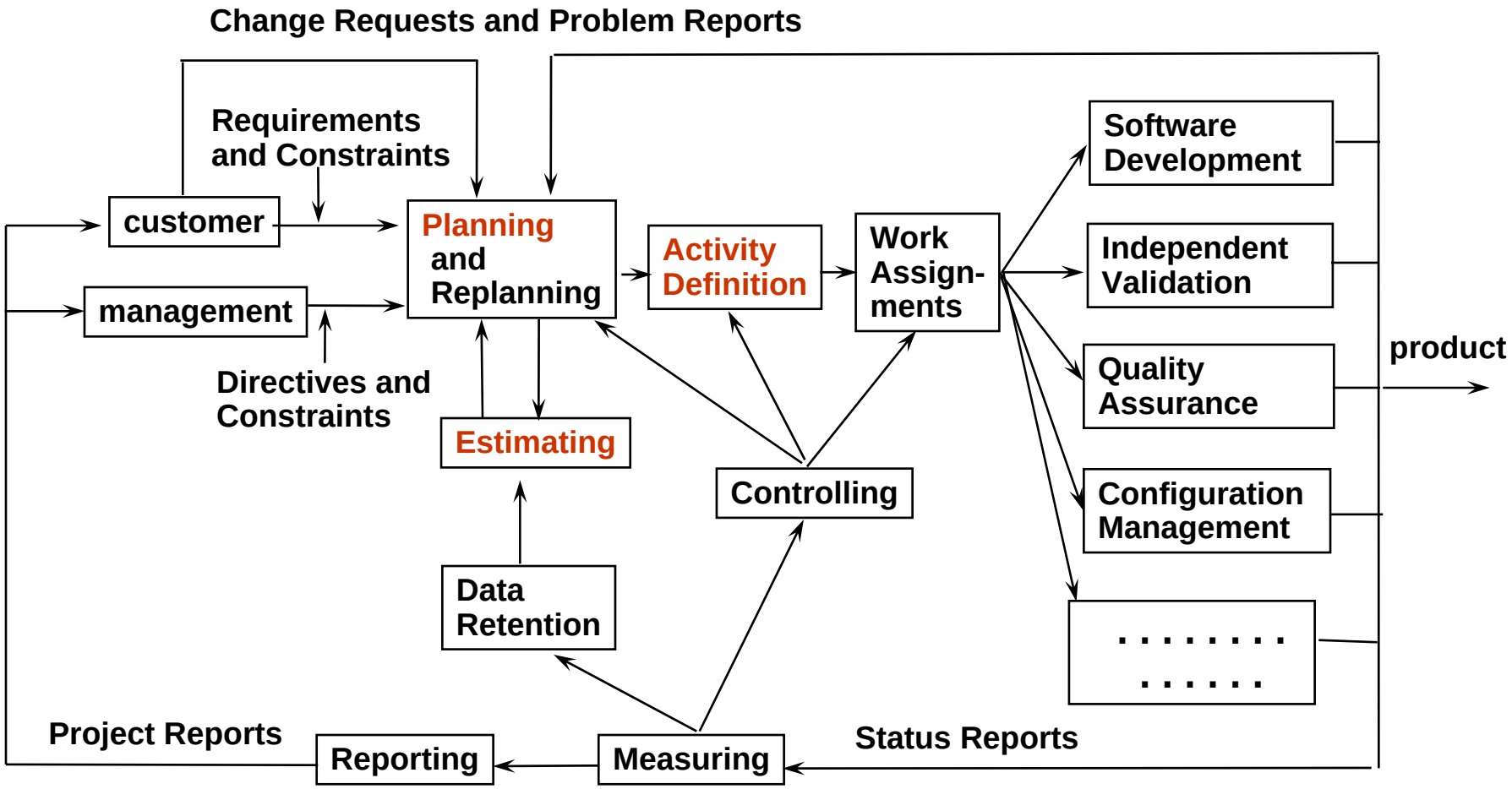
Additional Information

- The planning techniques presented in Chapter 5 are informed by:
  - the Project Planning process area of the CMMI-DEV-v1.2 process framework,
  - the planning elements of ISO and IEEE Standards 12207,
  - IEEE Standard 1058, and
  - the PMI Body of Knowledge.
- These elements are described in Appendix 5A to Chapter 5 of the textbook.

Objectives for Chapter 5

- After reading this chapter and completing the exercises, you should understand:
  - the scope of planning
  - rolling wave planning
  - scenarios for developing a project plan
  - developing an architecture decomposition view
  - developing a work breakdown structure
  - developing the project schedule
  - developing resource profiles
  - resource Gantt charts
  - estimating project cost

A Workflow Model with Emphasis on Planning, Estimating, and Activity Definition for Software Projects



Scenarios for Developing a Project Plan

1. Given a set of operational requirements and constraints on one or more of the schedule, budget, and resources determine the feasibility of the project and needed values for the unconstrained elements
2. Given a list of features and quality attributes estimate and then commit to the schedule, budget, and resources needed to develop a system or product having those features and quality attributes.
3. Given a completion date and a budget, determine the characteristics of a product that can be built or modified within the constraints of specified time and money.

Note

- Regardless of the scenario for developing a project plan, the initial project plan must achieve compatibility among:
  - requirements,
  - schedule,
  - budget,
  - resources, and
  - technology.
- Subsequent revisions to the project plan must maintain this balance as requirements and other factors change.

Dictated Plans

- You may be given a set of requirements and told to implement the system within strict constraints of schedule and budget
- You may be handed a set of changes to be made along with a schedule and a budget and be given the responsibility of managing a project to make the modifications.

"dictated plans" typically have high rates of failure to deliver accepted products on schedule and within budget

Rolling Wave Planning (1)

- Rolling wave planning acknowledges that it is impossible to develop plans at the level of detail indicated throughout Chapter 5 during the initial planning phase of your software projects.
- When you are conducting a project, a recommended approach is to augment the high-level master plan with detailed plans for:
  - the coming month,
  - for the subsequent month, and
  - for three months hence.
- Each month the plans are moved forward one month; i.e. moved forward in a rolling-wave manner.
  - The plans for the next month should be detailed and specific.
  - The plans for two and three months hence should be as specific as possible.

Rolling Wave Planning (2)

- Rolling wave planning is illustrated in Figure 5.2 of the textbook:

  months:

  n n+1 n+2

  n+1 n+2 n+3

  n+2 n+3 n+4

The Project Planning Strategy

- The first pass at planning should be done without regard to constraints
  - the first pass forms the basis for making adjustments and tradeoffs to satisfy constraints
  - or for determining that the project, as constrained, is infeasible

Project Planning Tools and Techniques

- Architecture Decomposition View
- Work Breakdown Structures
- Work Packages
- Activity Networks
- CPM, PERT, and Gantt charts
- Resource Loading Histograms

CPM: Critical Path Method
PERT: Program Evaluation and Review Technique

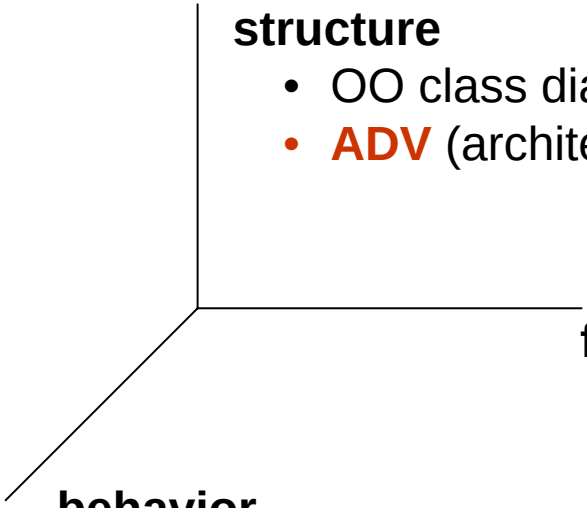
Initial Planning Activities (1)

- Developing a plan includes the following activities:.
  1. Develop an Architecture Decomposition View of the product architecture (ADV) and allocate requirements and interfaces to the elements of the ADV
  2. Develop a work breakdown structure that includes work elements for the ADV modules with allocated requirements and interfaces for each element of work
  3. Develop work packages for the tasks in the WBS
  4. Define a schedule of objectively measurable milestones
  5. Prepare a schedule network and identify the critical path (or paths)
  6. Prepare a PERT estimate of project duration

Initial Planning Activities (2)

7. Identify number and kinds of resources needed, when they will be needed, and for how long
8. Prepare an estimate of optimal effort, cost, schedule, and resources
9. Negotiate with the customer to obtain a balance among requirements, cost, and project duration that satisfies the project constraints

Three Views of Software and Some Notations Used



structure
- OO class diagrams
- **ADV** (architecture decomposition view)

function
- OO class methods
- data flow diagrams

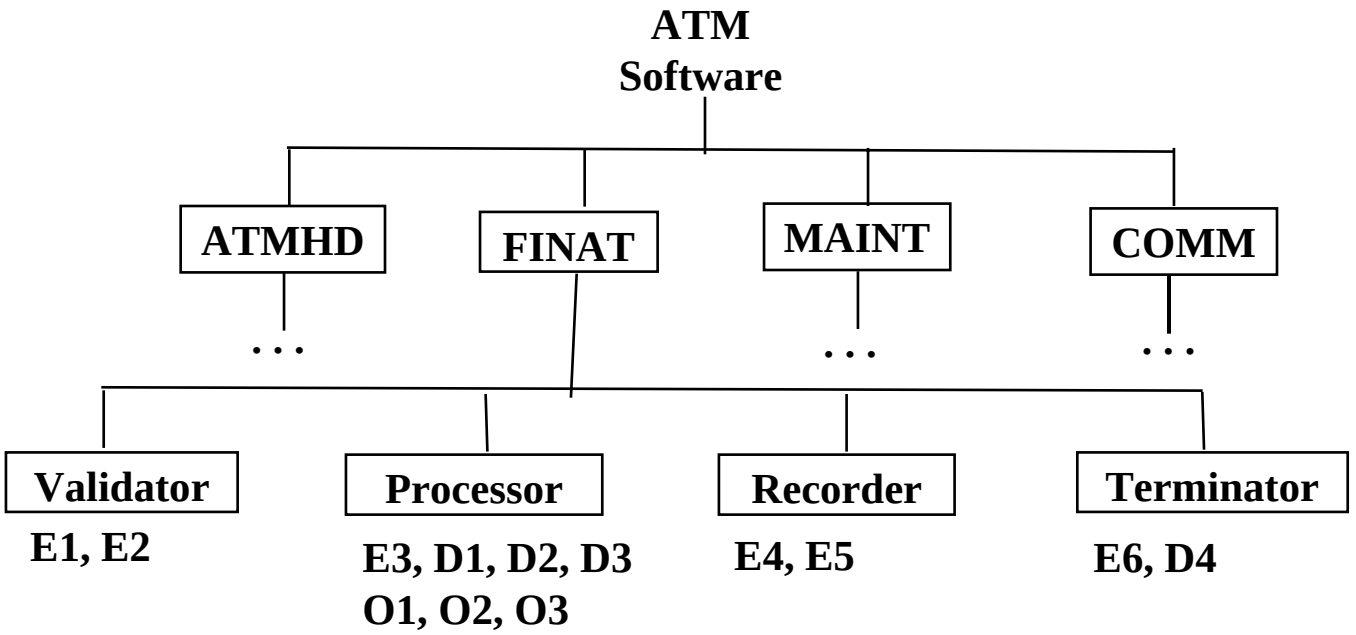
behavior
- state diagrams
- sequence diagrams

the ADV is the project manager's view of a product

ADV of an ATM

- Suppose we decide that software for ATM machines will have 4 major subsystems:
    - ATMHD: Hardware Drivers
    - FINAT: Financial Transactions
    - MAINT: Maintenance & Diagnostics
    - COMM: Communication Package
- Also, suppose FINAT is designed to have the following components:
    - Validator
    - Processor
    - Recorder
    - Terminator

Partial Architecture Decomposition View of ATM Software with Allocated Requirements



ATMHD: Hardware Drivers
FINAT: Financial Transactions
MAINT: Maintenance & Diagnostics
COMM: Communications Package

Ei: an Essential requirement
Di: a Desirable requirement
Oi: an Optional requirement

Outline Representation of the Partial ADV

ATM Software

1. ATMHD
2. FINAT
    - 2.1. Validator [E1,E2]
    - 2.2. Processor [E3, D1, D2, D3, O1, O2, O3]
    - 2.3. Recorder [E4, E5]
    - 2.4. Terminator [E6, D4]
3. MAINT
4. COMM

Design Guidelines for ADVs

Guidelines for designing an ADV include:

- limit the breadth (fan-out) to 7 or less at each level
- limit the depth of *each ADV** to 4 or 5 levels
- use a decimal numbering system to indicate the membership of each element of the ADV
- trace allocated requirements to ADV elements
- design the product structure to facilitate work assignments

* a large, complex system may have multiple subsystems, each having its own ADV

Traceability of Functional Requirements

	Validator	Processor	Recorder	Terminator
E1	X			
E2	X			
E3		X		
E4			X	
E5			X	
E6				X
D1		X		
D2		X		
D3		X		
D4				X
O1		X		
O2		X		
O3		X		

- Note that each functional requirement is allocated to one, and only one, element of the FINAT ADV

Traceability of Quality Attributes

- Some quality attributes may apply to:
  - an individual element of the ADV
  - several elements of the ADV
  - all elements of the ADV

Designing the Product Structure to Facilitate Work Assignments

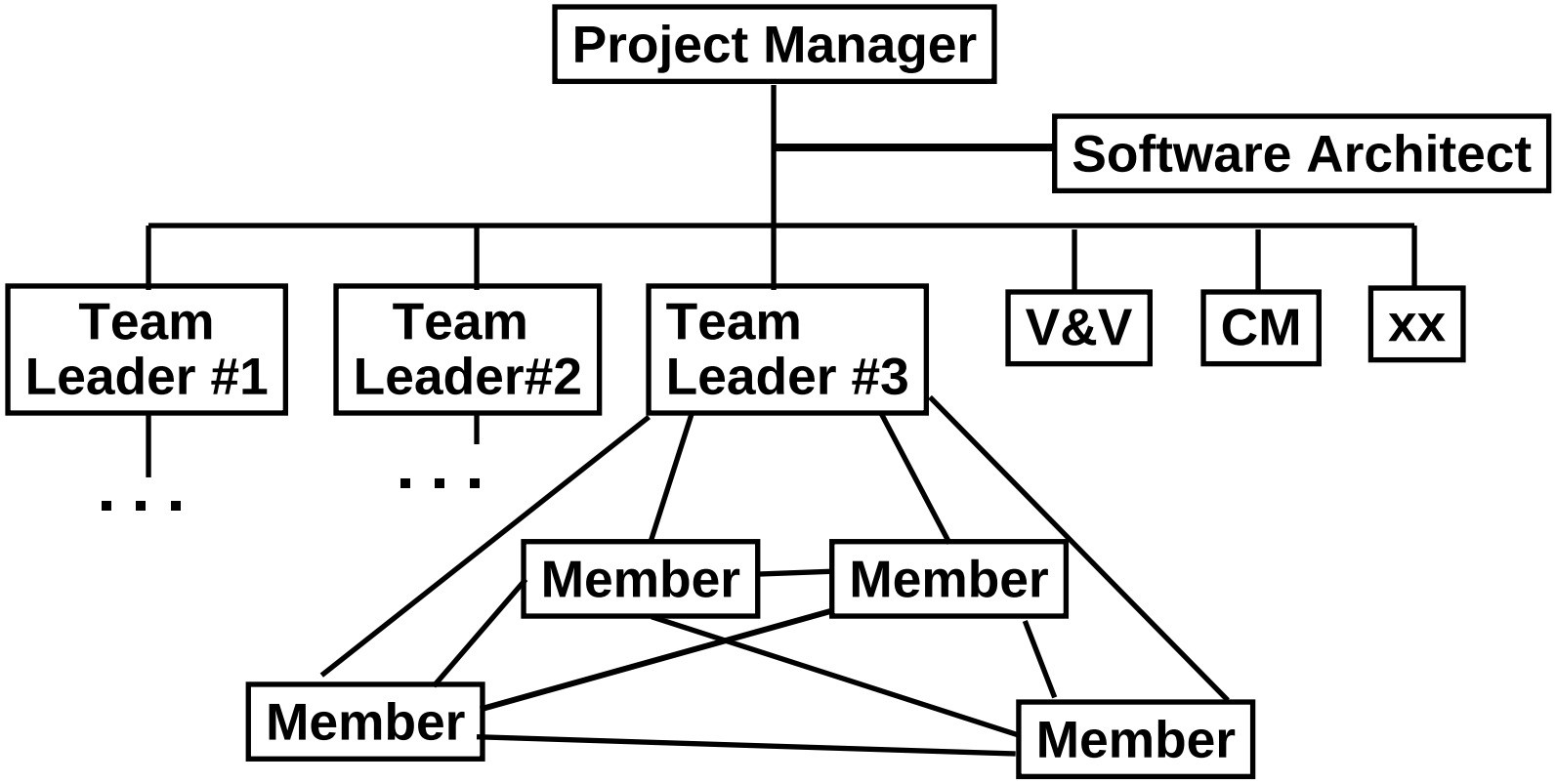
Conway's Law

> "The structure of a software system tends to resemble the structure of the team that builds it."

Fairley's Corollary

> "The architectural hierarchy of a software system must be structured to reflect the structure of work assignments for the team, or teams, that will build it."

A Structural Model for Software Projects

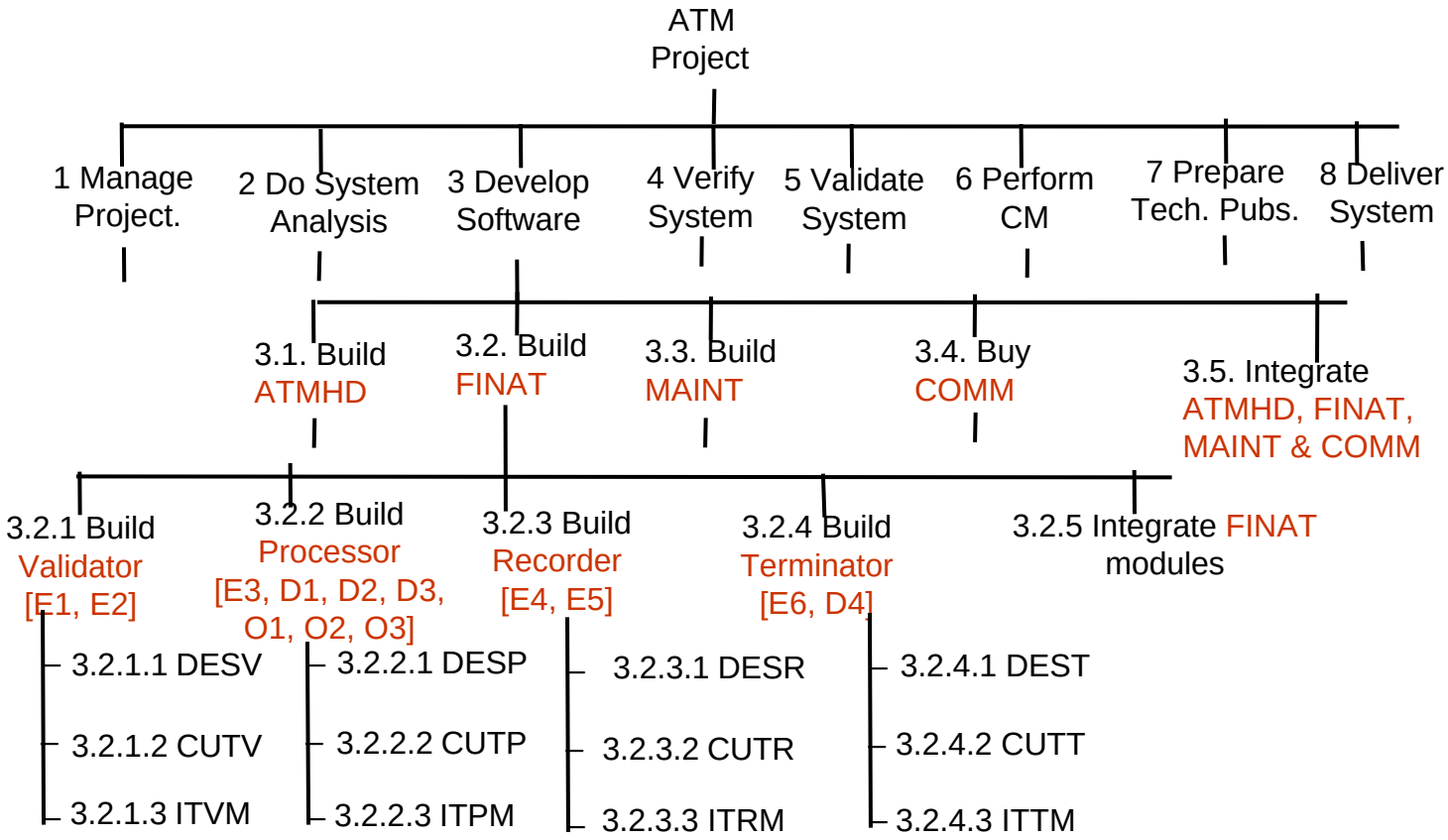


each team consists of 2 to 5 members

Work Breakdown Structures

- A Work Breakdown Structure (WBS) is a hierarchical decomposition of work activities
  - a WBS can be represented as a tree structure or an indented list
  - hierarchy partitions the development activities into separate parts
    - so that requirements and work activities can be assigned to different groups and different individuals

Graphical Form of a Partial WBS with Embedded ADV and Allocated Requirements



DESx: detailed design of module x; CUTx: coding & unit testing x; ITxC: integration and testing of x

Outline Form of the Partial WBS

1 Manage Project
2 Do System Analysis
3 Develop Software
  3.1 Build ATM Hardware Drivers
  3.2 Build Financial Transaction Handler
    3.2.1 Build Validator [E1, E2]
      3.2.1.1 Design Validator
      3.2.1.2 Code & Unit Test Validator
      3.2.1.3 Integrate & Test Validator
    3.2.2 Build Transaction Processor [E3, S1, D2, D3, O1, O2, O3]
      3.2.2.1 Design Transaction Processor
      3.2.2.2 Code & Unit Test Transaction Processor
      3.2.2.3 Integrate & Test Processor Components
    3.2.3 Build Recorder [E4, E5]
      3.2.3.1 Design Recorder
      3.2.3.2 Code & Unit Test Recorder
      3.2.3.3 Integrate & Test Recorder Module
    3.2.4 Build Terminator [E6,D4]
      3.2.4.1 Design Recorder
      3.2.4.2 Code & Unit Test Recorder
      3.2.4.3 Integrate & Test Recorder Module
  3.3 Build Maintenance & Diagnostic Module
  3.4 Buy the Communications Package
  3.5 Integrate ATMHD, FINAT, MAINT, and COMM modules
4 Verify System
5 Validate System
6 Perform CM
7 Prepare Technical Publications
8 Deliver System

WBS Elements for “Manage Project”

1 Manage Project
- 1.1 Initiate Project
  - 1.1.1 Identify stakeholders
  - 1.1.2 Develop/clarify requirements
  - 1.1.3 Prepare initial estimates
  - 1.1.4 Prepare initial project plan
  - 1.1.5 Obtain commitment to the plan
- 1.2 Conduct Project
  - 1.2.1 Measure and control project
  - 1.2.2 Lead and direct personnel
  - 1.2.3 Communicate and coordinate
  - 1.2.4 Manage risk
- 1.3 Closeout Project
  - 1.3.1 Obtain product acceptance
  - 1.3.2 Conduct post-mortem sessions
  - 1.3.3 Prepare and distribute lessons-learned report
  - 1.3.4 Assist in reassigning project personnel

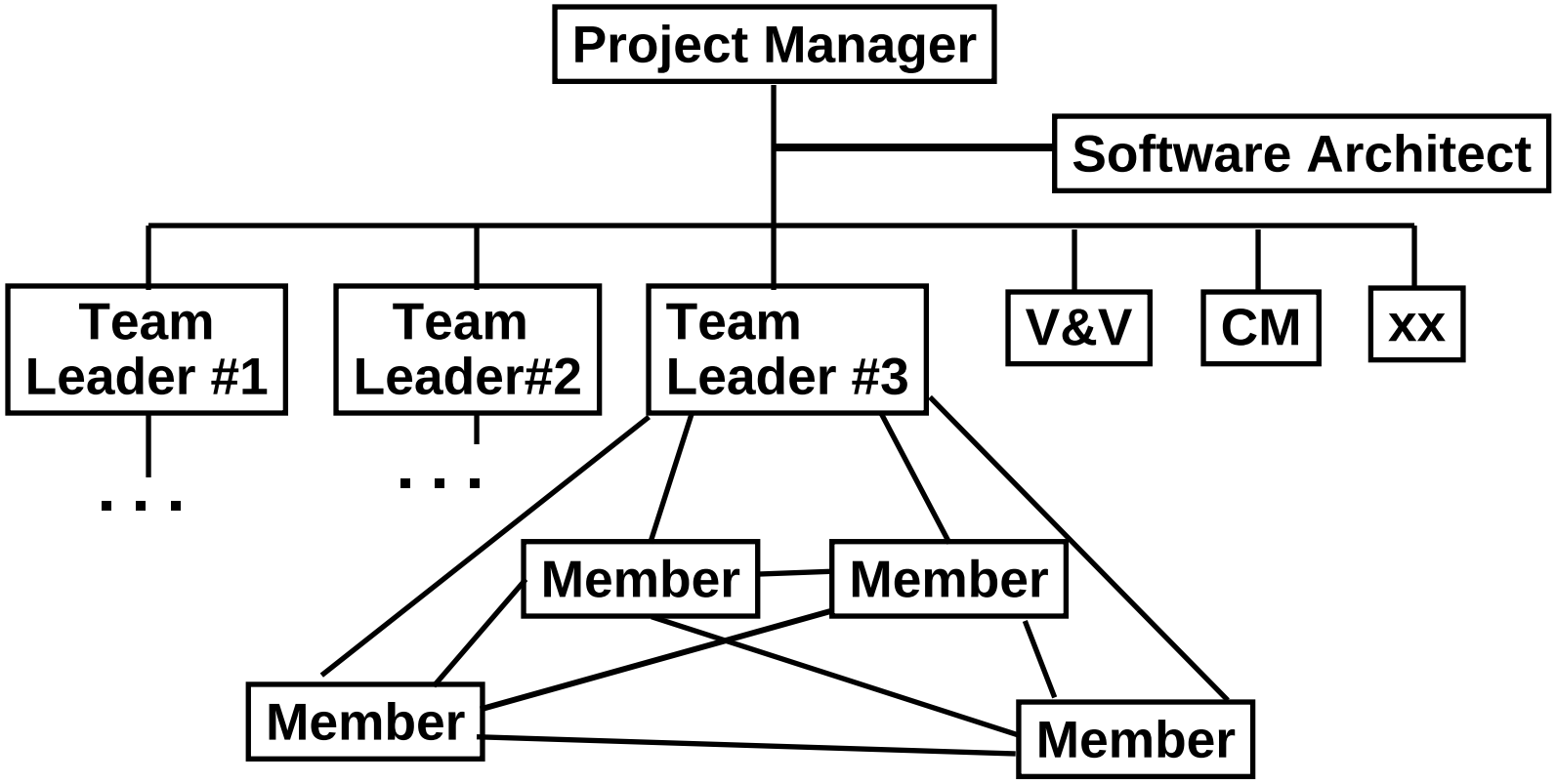
Notes

1. The ADV contains nouns to denote things
2. The WBS contains verb phrases to denote work activities
3. The work activities to develop the elements of the ADV are embedded in the WBS
   - the nouns in the ADV are converted to verb phrases in the WBS

     e.g., “Validator” in the ADV becomes

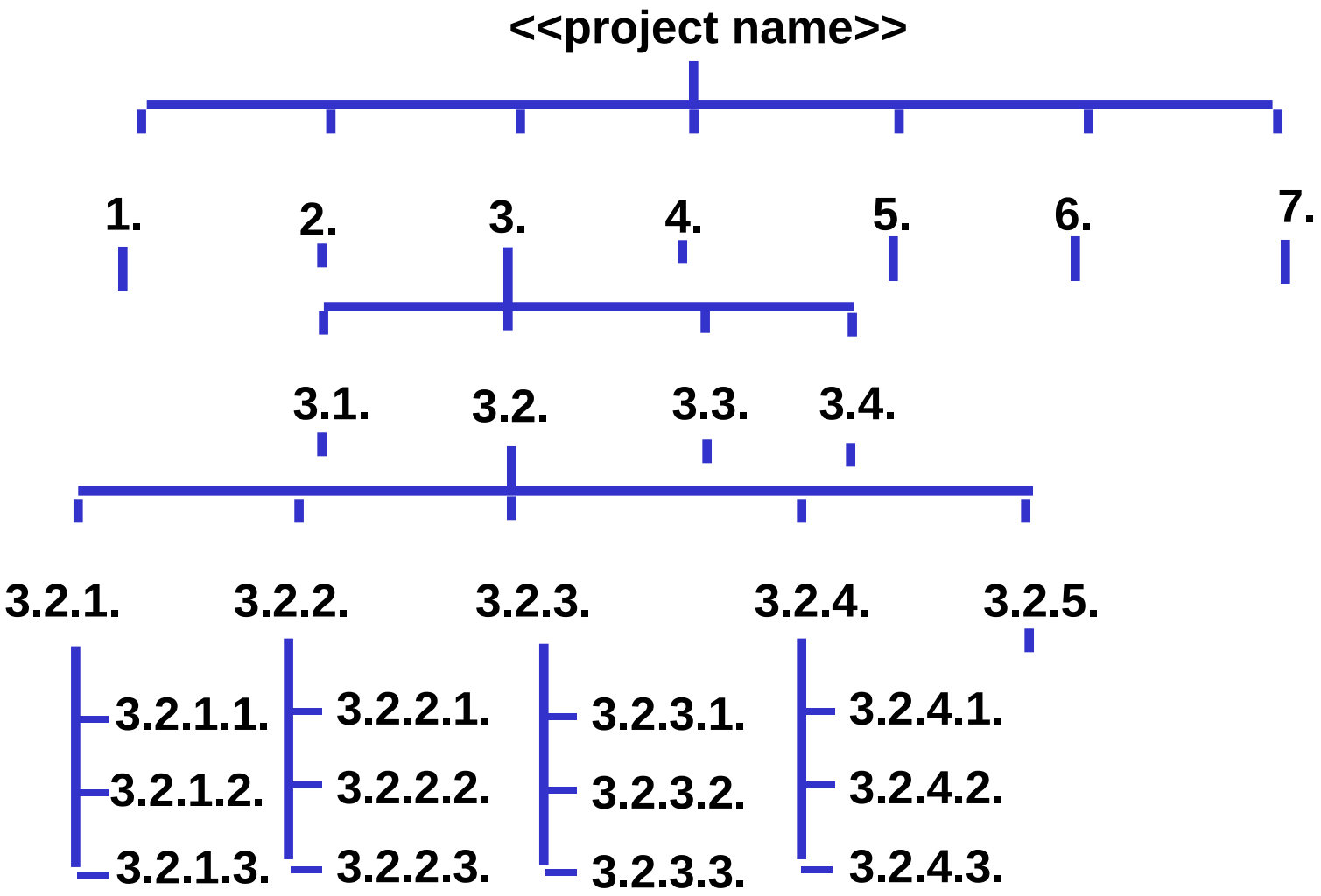
     “Build Validator” in the WBS

Mapping the WBS to the Organizational Structure of Software Projects



each team consists of 2 to 5 members

Structure & Numbering of a Partial WBS



Decomposition Criteria for the ADV and WBS

The initial versions of the ADV and WBS should be decomposed until:

1. hidden complexities are exposed (i.e., the job to be done is understood);
2. opportunities for reuse of existing software components can be identified;
3. the necessary hardware resources such as computer memory and processor speed can be specified, and
4. estimates of effort needed to develop the software can be made
5. A schedule can be developed

WBS Design Guidelines

Table 5.2 in Chapter 5 of the textbook lists 15 guidelines for designing work breakdown structures

Design Guidelines for Work Breakdown Structures

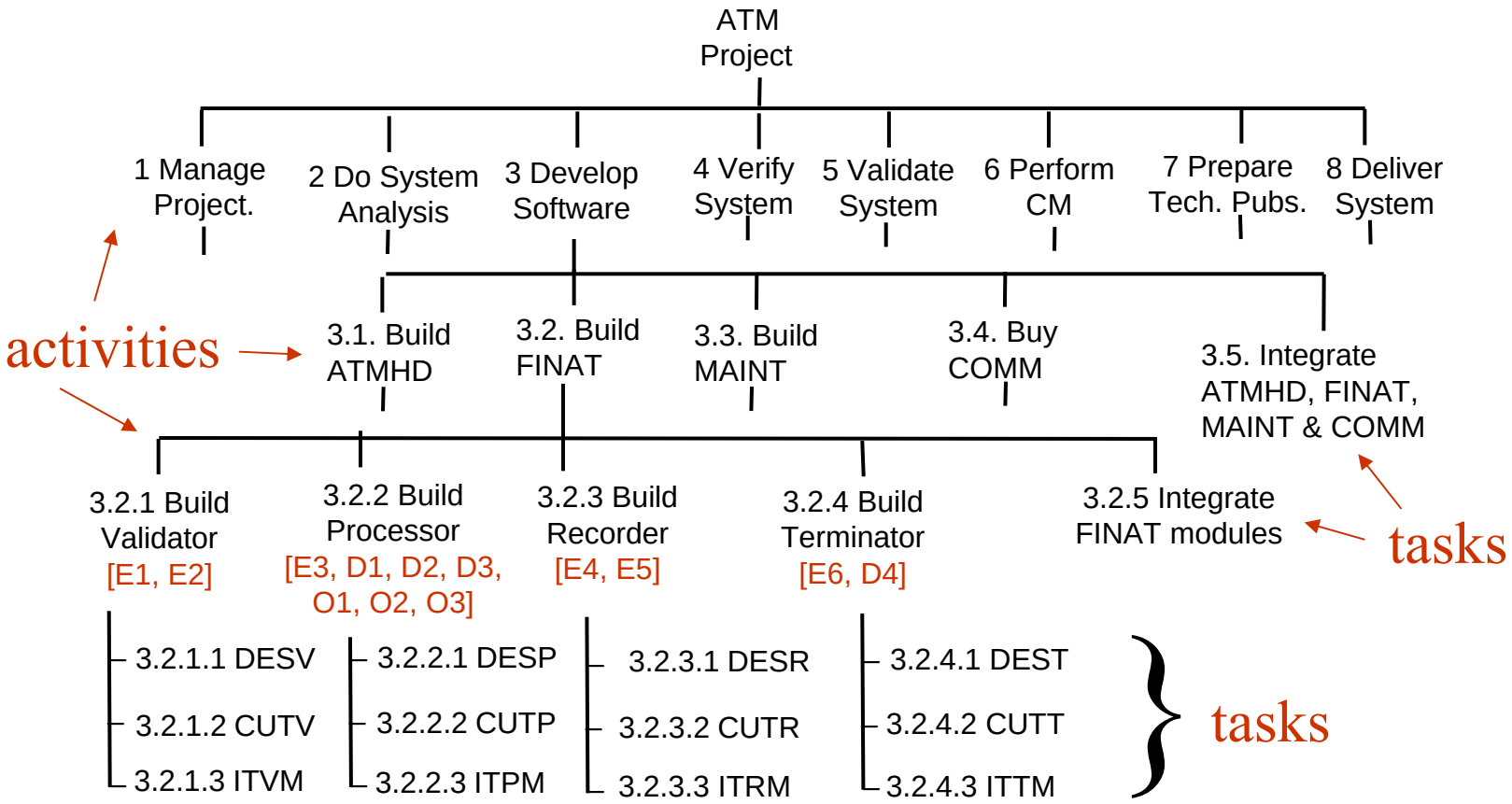
Some of the guidelines for designing a WBS include:

- name the elements using verb phrases
- limit the breadth (fan-out) at each level to 7 or 8
- limit the depth of each WBS to 4 or 5 levels
- embed the product structure in the WBS
- design the product structure to facilitate concurrent work assignments
- use a decimal numbering system to indicate the membership of each WBS element
- trace allocated requirements to WBS elements
- document each element of the WBS in a work package
- observe inherited constraints when designing lower-level work packages
- elaborate the WBS using the rolling-wave approach

Work Packages (1)

- The lowest level elements in a WBS are called ***tasks***
  - they are the smallest units of management planning and control
  - higher level units of work are called ***activities***
    - they are aggregations of lowest level tasks and subordinate activities
- Work packages are used to document tasks
  - work packages for activities are aggregations of the work packages for lowest level tasks and subordinate activities

Partial WBS for an ATM Project



Work Packages (2)

- A work package for a task should contain:
  - the corresponding WBS number and name
  - a brief description of the task
  - estimated duration
  - resources needed
  - predecessor and successor tasks
  - work products to be produced
  - work products that will be placed under version control (baselined)
  - risk factors (i.e., potential problems that might interfere with successful completion of the work package)
  - objective acceptance criteria for the work products generated by the task

A Work Package Template and Example

Task identifier: 3.2.2.1 Design Transaction Processor
Task description: Specify internal architecture of the Transaction Processor module
Estimated duration: 2 weeks
Resources needed:
Personnel: 2 senior telecom designers
Skills: Designers must know UML
Tools: One workstation running Rapsody
Travel: 3 day Design Review in San Diego for 2 people
Predecessor tasks: 3.2.1 - Develop system architecture
Successor tasks: 3.3.2.2 - Implement Transaction Processor
Work products: Architectural specification for Transaction Processor
Test Plan for Transaction Processor
Baselines created: Architectural Specification and Test Plan
Risk factors: Designers not identified
Acceptance criteria: Successful design inspection by peers and approval of Transaction Processor design by the Software Architect

Who Prepares the Work Packages?

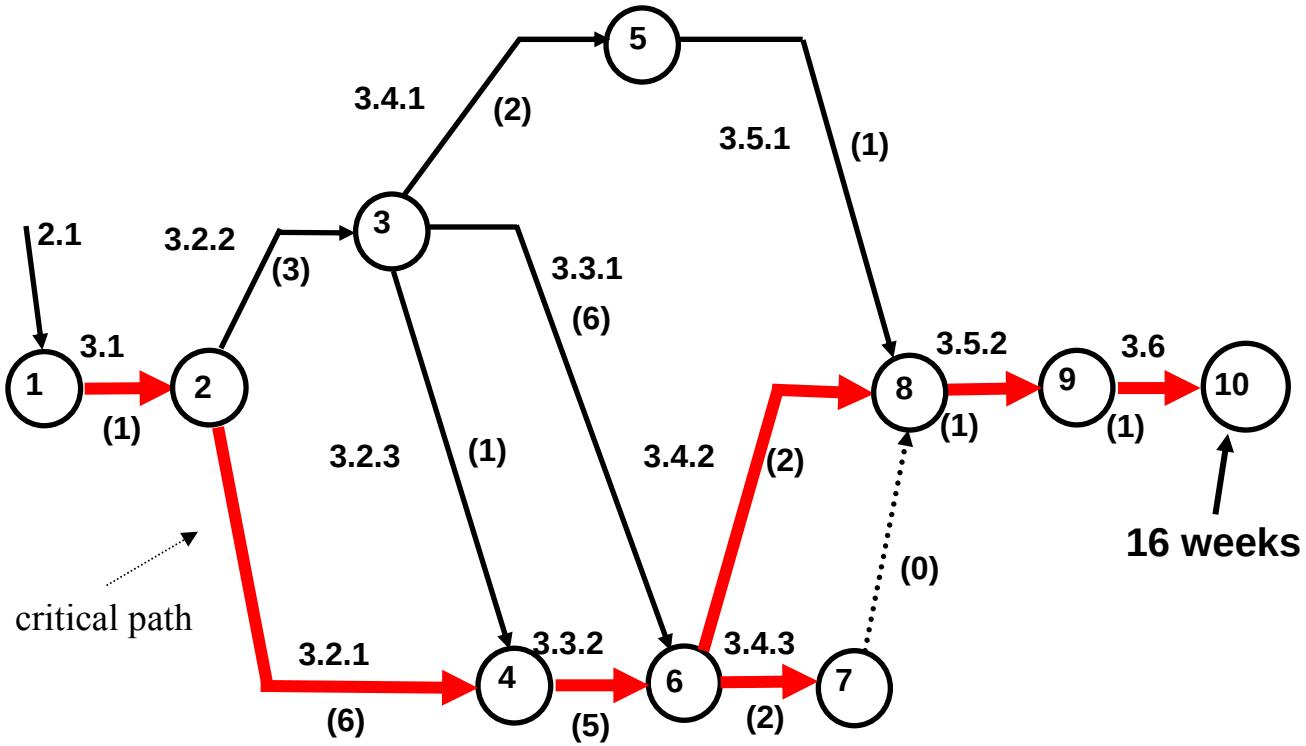
- During initial planning, the high level work packages are prepared by the planning person or team
- During project execution, each team leader works with his or her team members to prepare work packages on a rolling wave basis
- Team leaders are responsible for tracking and reporting progress on work packages
  - and they are responsible for the quality of the work generated by the team members

On a small project (fewer than 6 or 7 people) the project manager may be the architect and the team leader

A WBS Task List

Task #	Description	Predecessors	Duration	#Staff
2.1	Receive approval to proceed	--	--	--
3.1	Analyze requirements	2.1	1	2
3.2	Design			
3.2.1	Redesign existing components	3.1	6	4
3.2.2	Design new components	3.1	3	1
3.2.3	Design interfaces	3.2.2	1	2
3.3	Implement			
3.3.1	Implement new code	3.2.2	6	2
3.3.2	Modify existing code	3.2.1, 3.2.3	5	1
3.4	Finish implementation			
3.4.1	Develop integration plan	3.2.2	2	2
3.4.2	Finish unit testing	3.3.1, 3.3.2	2	2
3.4.3	Update documentation	3.3.1, 3.3.2	2	3
3.5	Integrate and test			
3.5.1	Develop integration tests	3.4.1	1	3
3.5.2	Perform integration tests	3.4.2&3, 3.5.1	1	2
3.6	Perform acceptance tests	3.5.2	1	1

A Critical-Path Schedule Network Generated from the Task List



m.n = tasks; (x) = activity duration

(n) = milestones;

Milestone Event List

Event	Description
1	Project initiation
2	Requirements analysis completed
3	Design of new components completed
4	Existing components redesigned
	Interfaces to new components designed
5	Integration plan completed
6	New code implemented
	Existing code modified
7	Documentation updated
8	Unit testing completed Documentation updated
	Integration tests ready
9	Integration tests completed
10	Acceptance tests completed

achievement of a milestone requires tangible evidence of work products completed

Critical-Path Milestone Chart

critical milestone	elapsed time	cumulative time
3.1	1 week	1 week
3.2	6 weeks	7 weeks
3.7	5 weeks	12 weeks
3.8, 3.9	2 weeks	14 weeks
3.11	1 week	15 weeks
3.12	1 week	16 weeks

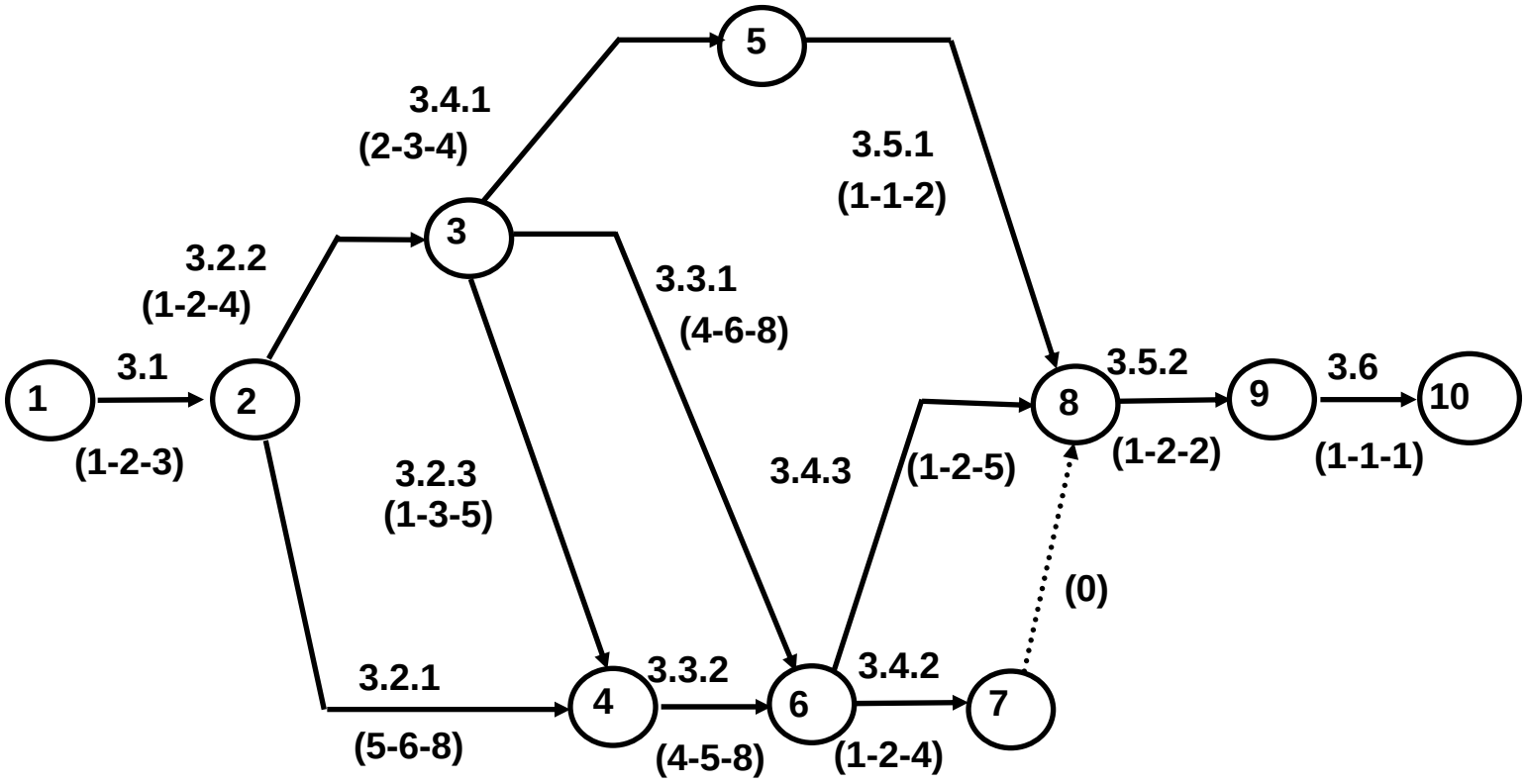
note: some weekly milestones should be inserted between 3.1 & 3.2 and between 3.2 & 3.7

An Observation

- A WBS imposes a tree-structured “is-part-of” relation on a collection of work activities and tasks
  - an aggregation relation
- A schedule network imposes an “is-preceded-by” relation on a collection of tasks
  - a time-ordering relation
- The WBS contains no sequencing (scheduling) information
- The activity network contains no hierarchical (aggregation) information

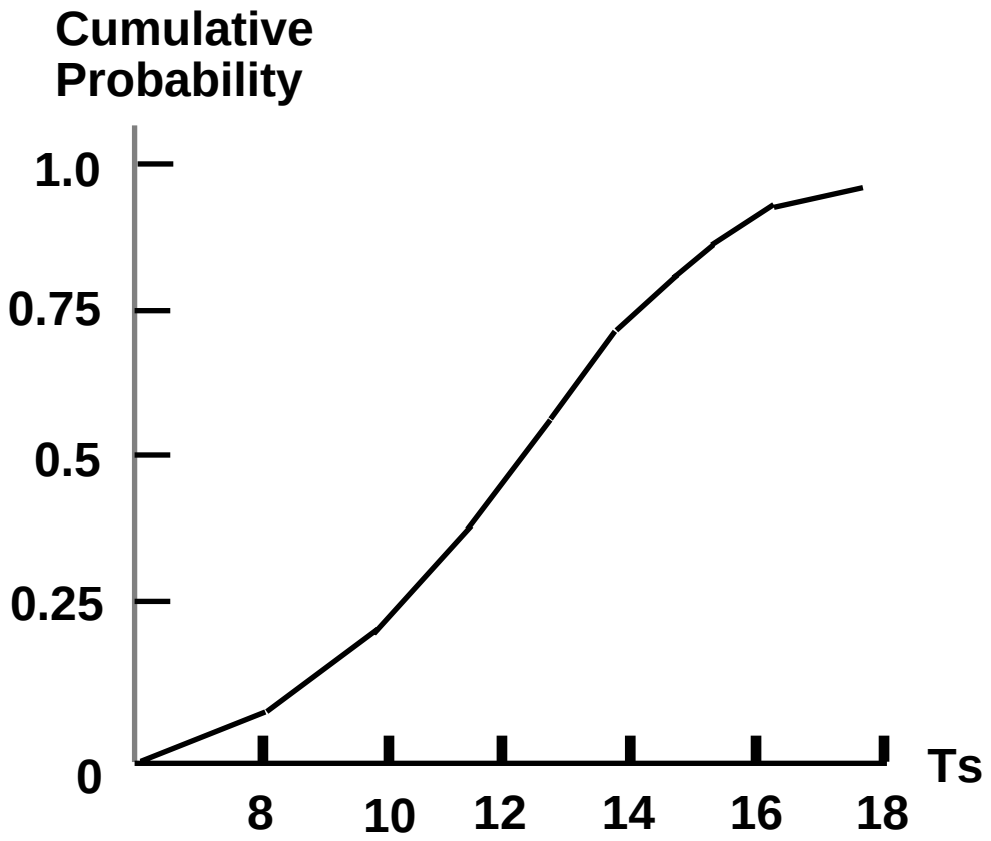
both views are necessary

A PERT Network



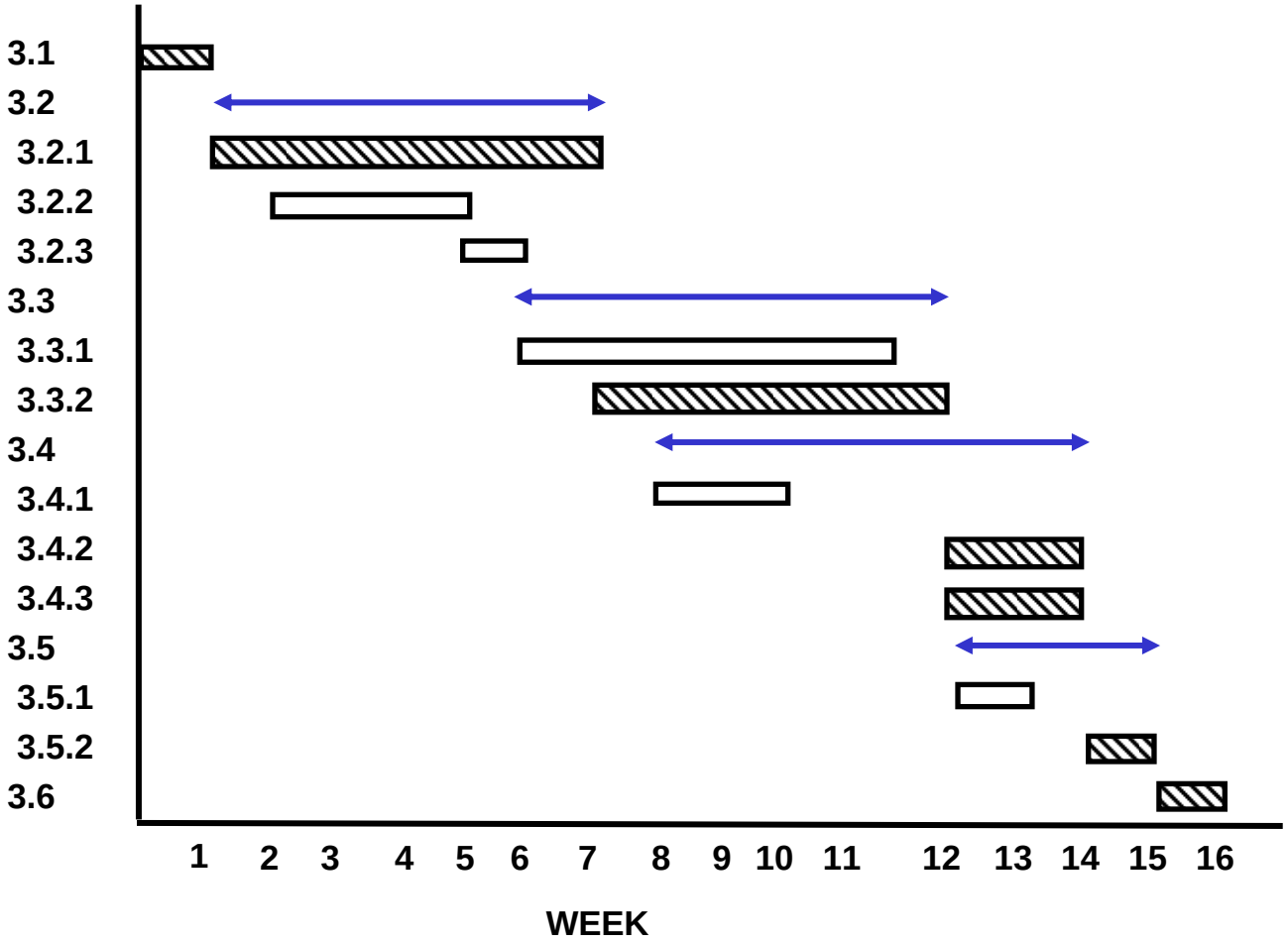
m.n = activities; (n) = milestones;
(a-m-b) = activity duration estimates

A Cumulative PERT Probability Distribution for a Schedule

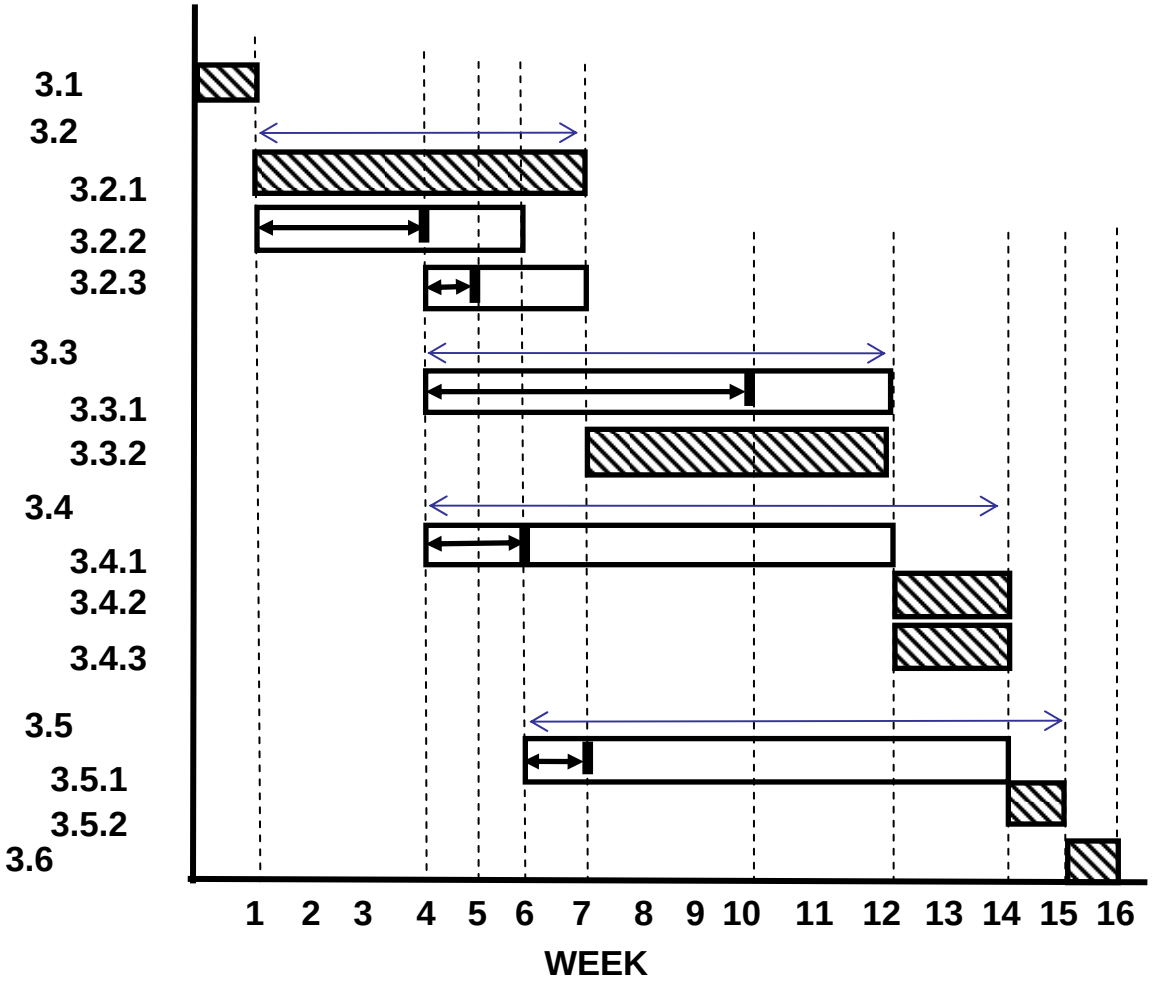


Probability of Completion in time t $\leq$ Ts
(probability of achieving milestone 10)

A WBS – Gantt Chart

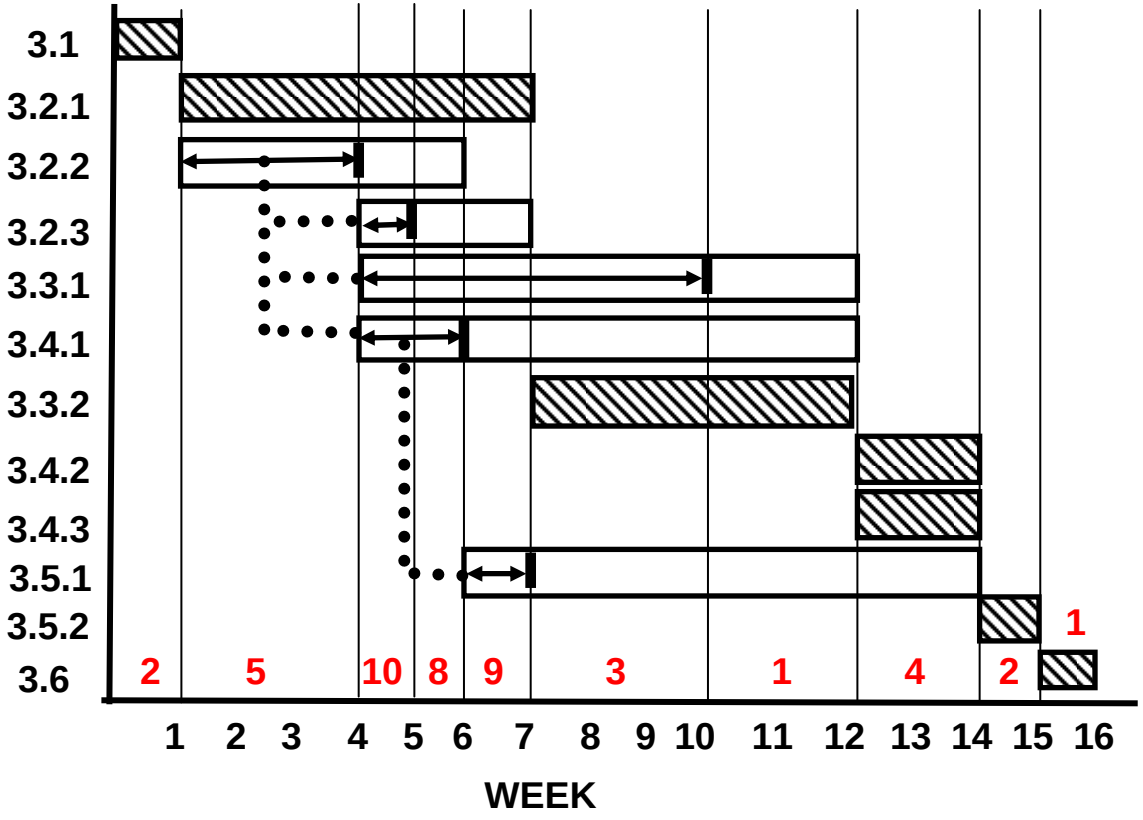


A Gantt Chart with Slack Times

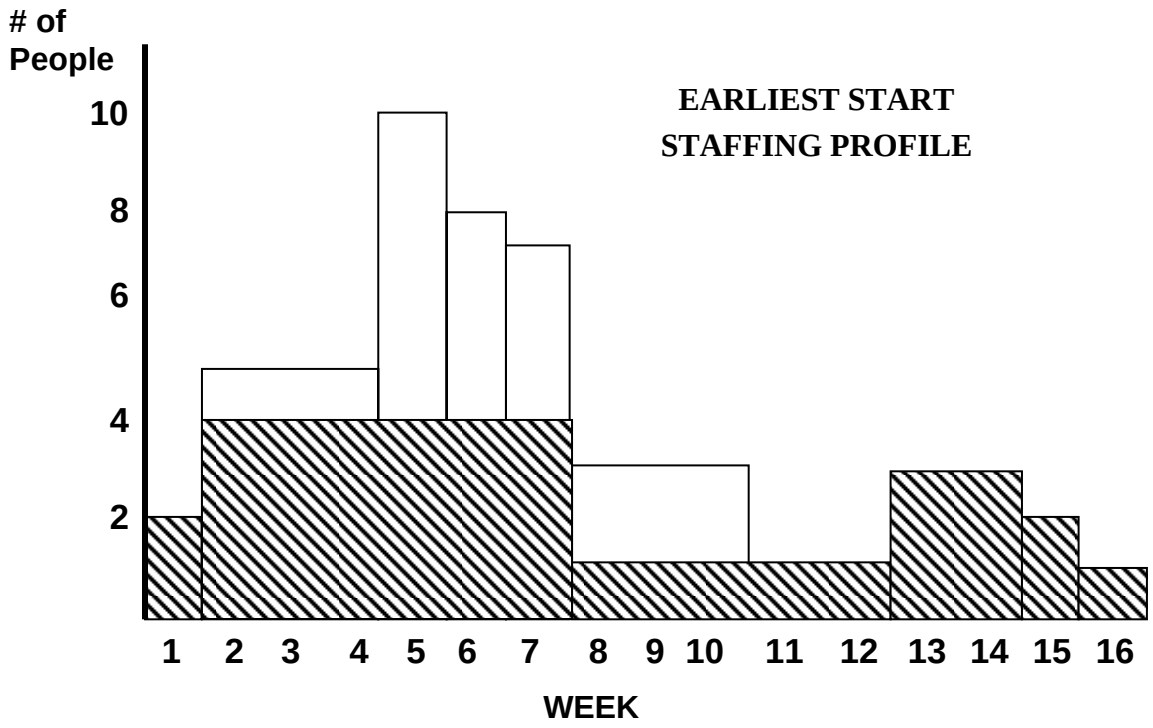


cross-hatched boxes indicate critical path tasks
clear boxes indicate tasks with slack times

A Linked-Task Gantt Chart with # of Personnel Needed

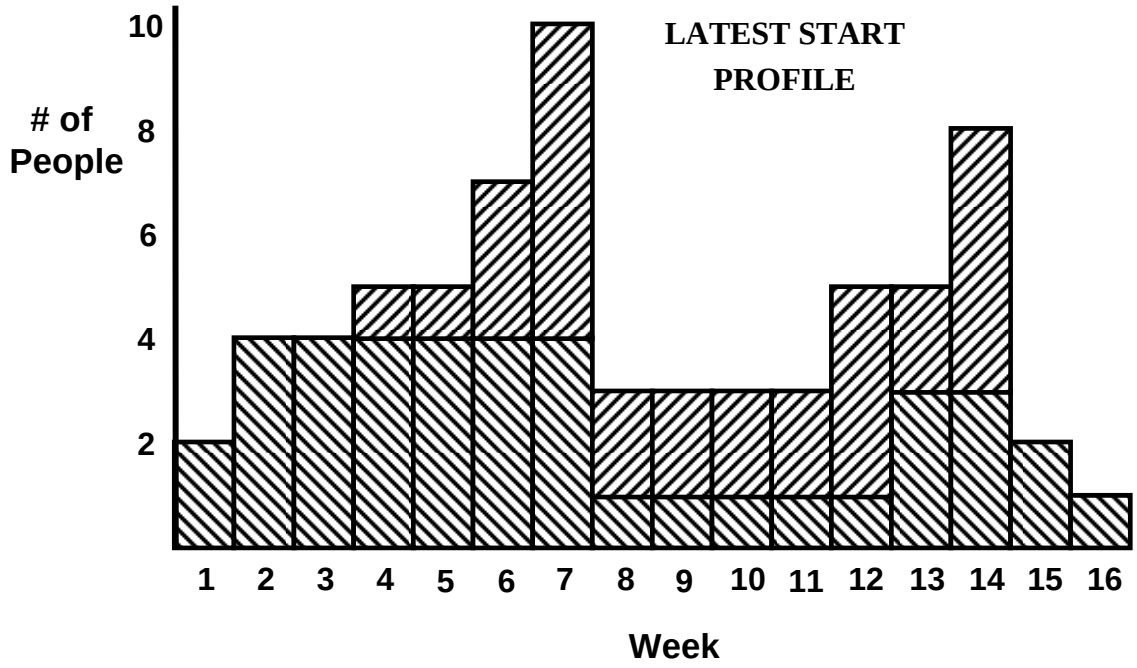


Earliest Start Time Staffing Profile



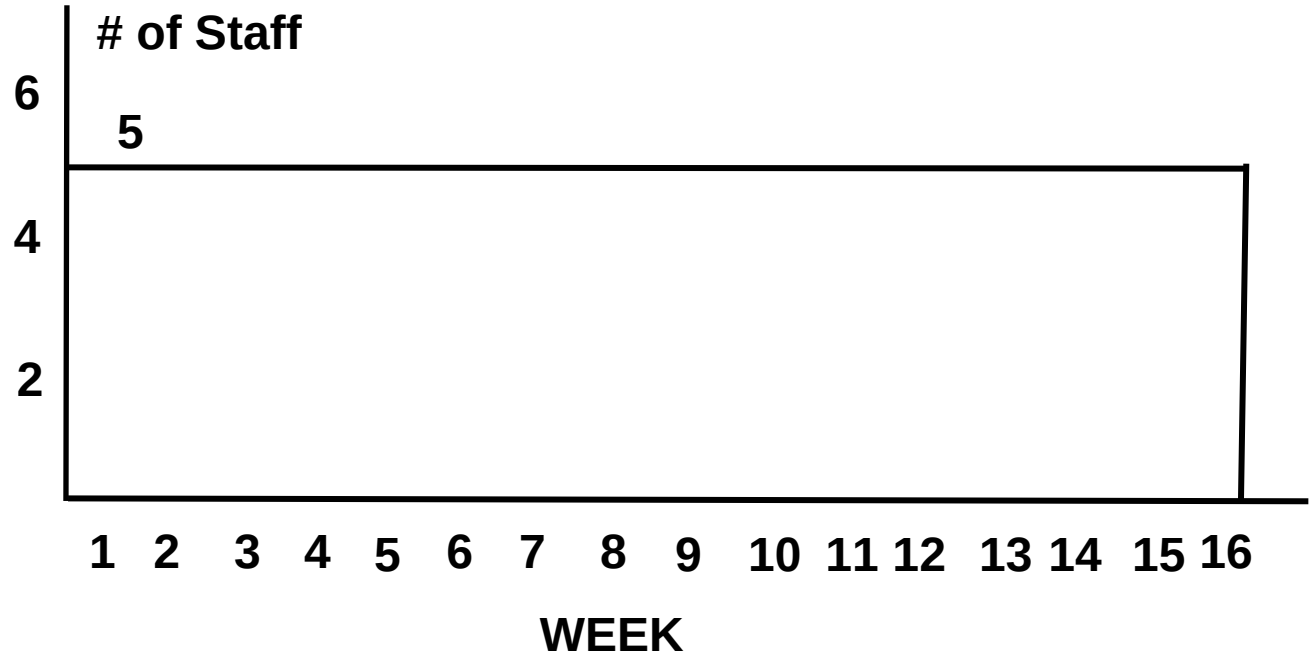
cross-hatched boxes indicate critical path tasks
clear boxes indicate tasks with slack times

Latest Start Time Staffing Profile



cross-hatched boxes indicate all tasks are on critical paths

A Desirable, but Unobtainable Schedule for the Example



The schedule/resource allocation problem in this example is caused by the large number of software developers needed in weeks 4, 5, and 6

Achieving a Schedule – Resources Balance (1)

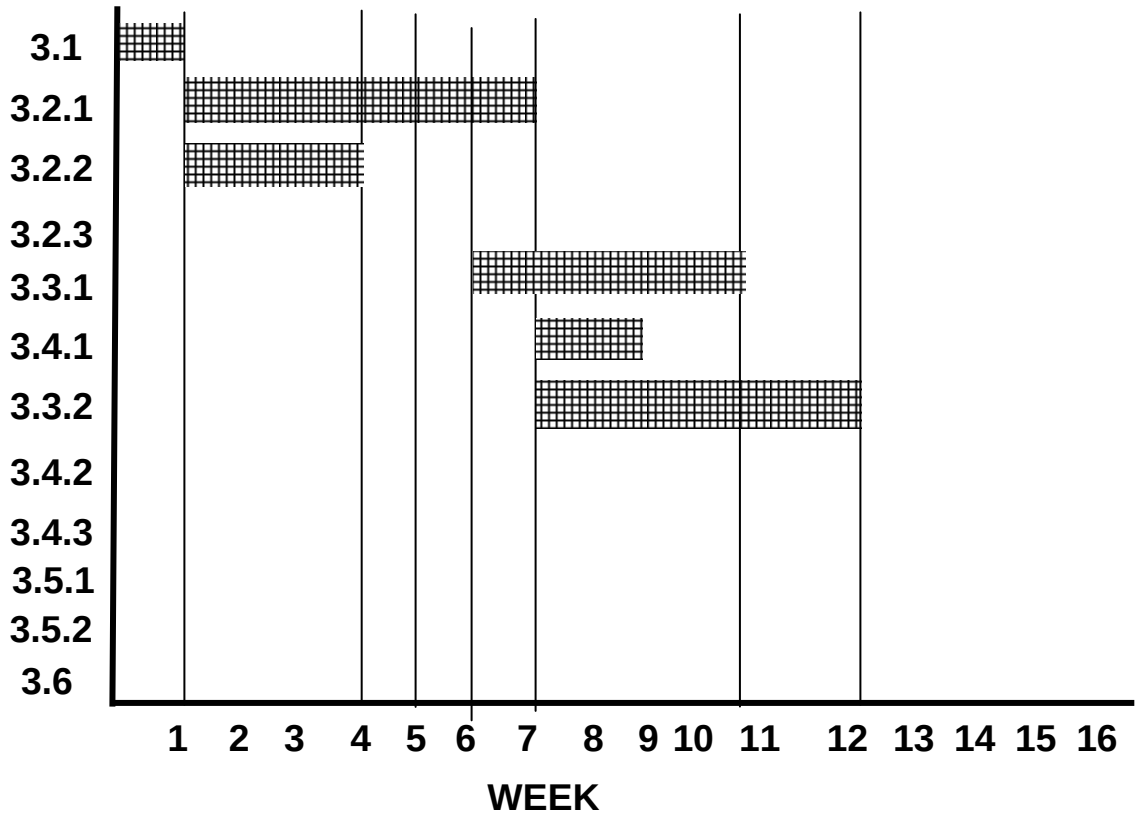
- Acceptable options for achieving a balance include:
  - extending the schedule so that fewer resources are needed in peak weeks
  - adding more resources to maintain the schedule
  - using more productive resources so that fewer numbers are needed
  - descoping the requirements so that fewer resources and less time are needed
  - rearranging the tasks so that fewer resources are needed in peak weeks

combinations of these options can be used to achieve an acceptable balance

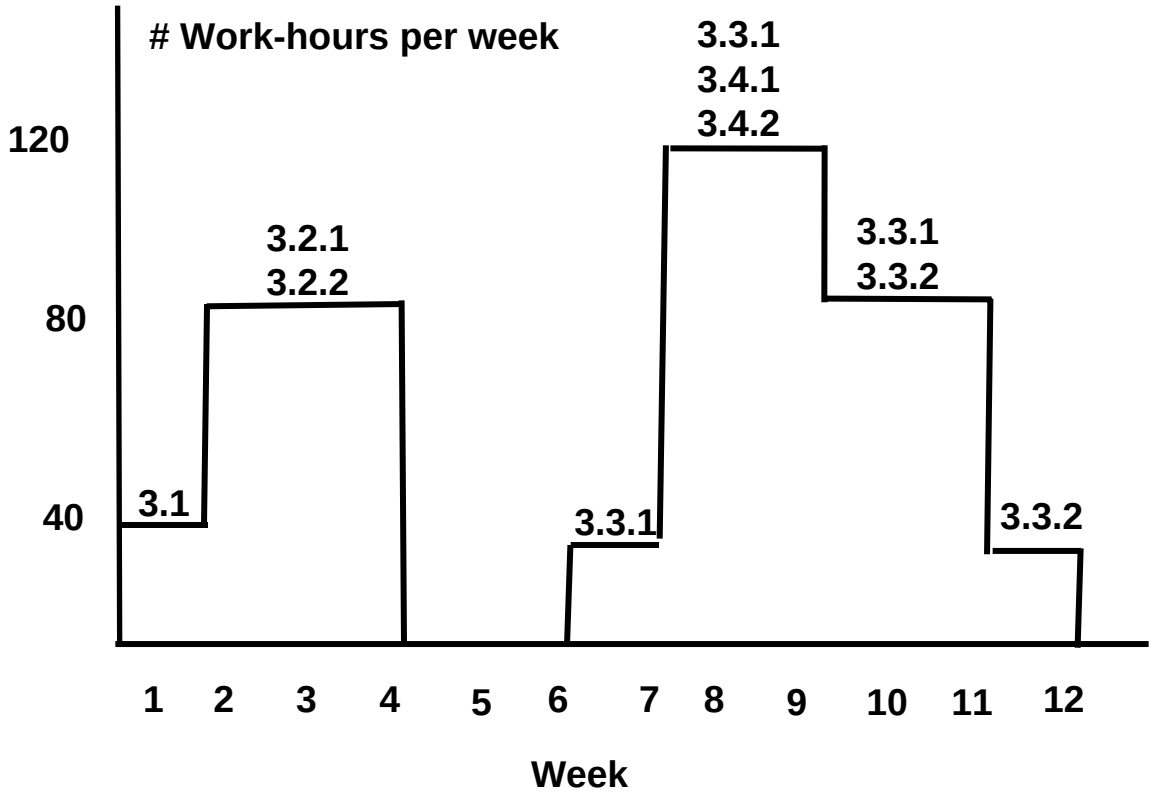
Achieving a Schedule – Resources Balance (2)

- Unacceptable options include:
  - producing an unrealistic plan that has no chance of being successfully implemented
  - reducing or eliminating quality control activities such as inspections, reviews, and testing
  - planning for overtime

A Resource Gantt Chart for Joe Hotshot



An Infeasible Staffing Profile for Scarce Resource Joe Hotshot



A WBS Task List with Calculated Effort

Task #	Description	Predecessors	Duration	#Staff	Effort
2.1	Receive approval to proceed	--	--	--	
3.1	Analyze requirements	2.1	1	2	2
3.2	Design				
3.2.1	Redesign existing components	3.1	6	4	24
3.2.2	Design new components	3.1	3	1	3
3.2.3	Design interfaces	3.2.2	1	2	2
3.3	Implement code				
3.3.1	Implement new code	3.2.2	6	2	12
3.3.2	Modify existing code	3.2.1, 3.2.3	5	1	5
3.4	Finish implementation				
3.4.1	Develop integration plan	3.2.2	2	2	4
3.4.2	Finish unit testing	3.3.1, 3.3.2	2	2	4
3.4.3	Update documentation	3.3.1, 3.3.2	2	3	6
3.5	Integrate and test				
3.5.1	Develop integration tests	3.4.1	1	3	3
3.5.2	Perform integration tests	3.4.2&3, 3.5.1	1	2	2
3.6	Perform acceptance tests	3.5.2	1	1	1

Effort: Σ(Duration x #Staff) = **68 staff-weeks**

Estimating Effort, Cost, and Schedule

- If loaded salary per staff-week is X and the 68 staff-weeks represent 50% of project cost, the estimated cost is 2*X*68
  - If, for example, loaded salaries* are $2500 US per week, the cost of the project is estimated to be $340,000 US
  - The critical path approach indicates that the project will require 16 weeks or more
    - perhaps more to account for scheduling constraints of scarce resources
  - The PERT example indicates that the project can be completed in 15 weeks or less at 85% probability, subject to resource availability constraints
- Additional techniques for estimating effort, schedule, resources, and cost are presented in Chapter 6

* loaded salaries: the cost of employees for the organization

The Main Points of Chapter 5 (1)

- Project plans must be consistent with product requirements; you cannot prepare a plan for developing a software product if you don't know what product to make
- The more you understand about the product to be made, the more confident you will be in the details of your plan
- A project plan must be updated periodically and as events dictate using a rolling wave approach
- Your initial plan and subsequent plans must maintain a balance among requirements, schedule, budget, and resource availability
- Essential elements of a project plan include a WBS, an activity network, resource profiles for the various kinds or resources, and strategies for dealing with identified risk factors
- The Work Breakdown Structure (WBS) is a fundamental tool for planning, tracking, and controlling a software project
- The Architecture Decomposition View (ADV) of the software architecture provides the basis for developing a WBS

The Main Points of Chapter 5 (2)

- The ADV is product-oriented; nouns are used to specify things
- The WBS is process-oriented; verb phrases are used to specify activities and tasks
- Using the 15 guidelines for designing a WBS will ensure that the WBS is designed with the same care that is used to design the product
- Your initial WBS should be decomposed to satisfy the WBS decomposition criteria
- Work packages are the specifications for tasks and activities in the WBS
- Work packages for activities are aggregations of work packages for subordinate tasks and activities
- The schedule network, resource requirements, cost estimates, and risk factors can be derived from work packages
- The Critical Path Method (CPM) can be used to determine the minimum estimated duration of a project and the slack times associated with non-critical tasks

The Main Points of Chapter 5 (3)

- The Program Evaluation and Review Technique (PERT) can be used to determine the times, at various levels of probability, required to reach project milestones, including the final milestone
- A task Gantt chart can be used to depict the critical path, illustrate slack times for non-critical tasks, and determine resource profiles for the various kinds of resources
- A resource Gantt chart can be used to illustrate the resource loading for various resources
- Resource profiles can be used to calculate effort and the costs of the various resources; project schedule can be determined from the critical path or from PERT calculations
- SEI, ISO, IEEE, and PMI provide frameworks, standards, and guidelines for project planning techniques
  - see Appendix 5A to Chapter 5

The Main Points of Chapter 5 (4)

- Acceptable options for reconciling schedule/resource conflicts include:
  - reconfiguring the schedule network
  - extending the schedule so that fewer resources are needed in peak weeks
  - adding more resources to maintain the schedule
  - using more productive resources so that fewer numbers are needed
  - descoping the requirements so that fewer resources and less time are needed
  - combinations of the above

The Main Points of Chapter 5 (5)

- Unacceptable options for reconciling schedule/resource conflicts include:
  - producing an unrealistic plan that has no chance of being successfully implemented
  - planning for overtime
  - reducing or eliminating quality control factors such as inspections, reviews, and testing