Software Project Scheduling

- Introduction
- Project scheduling
- Task network
- Timeline chart
- Earned value analysis

Introduction

Objectives of Planning and Scheduling

- Develop the necessary understanding and skills to produce and manage a simple project schedule.

Project Planning and Scheduling

- So far we have a project for which we have estimated duration and effort based on a high level understanding of what needs to be done.
- The majority of projects are 'completed' late, if at all.
- A project schedule is required to ensure that required project commitments are met.
- A schedule is required to track progress toward achieving these commitments.

Why Software Is Delivered Late?

Eight Reasons for Late Software Delivery

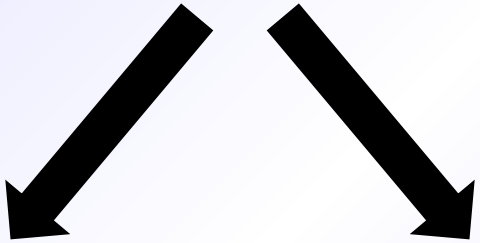
- An unrealistic deadline established by someone outside the software engineering group and forced on managers and practitioners within the group
- Changing customer requirements that are not reflected in schedule changes
- An honest underestimate of the amount of effort and /or the number of resources that will be required to do the job
- Predictable and/or unpredictable risks that were not considered when the project commenced
- Technical difficulties that could not have been foreseen in advance
- Human difficulties that could not have been foreseen in advance
- Miscommunication among project staff that results in delays
- A failure by project management to recognize that the project is falling behind schedule and a lack of action to correct the problem

Project Planning and Scheduling

"One day a time"

- All technical projects involve 100s of small tasks
- Some tasks do no affect the project completion
- Other tasks are *critical* for project completion
- Project manager must:
  - **define all project tasks**
  - **build a network that depicts their interdependence**
  - **identify the *critical* tasks**
  - **track the progress of these tasks**
  - **recognize the delay "*one day at a time*"**

Two different Perspectives to view Scheduling



End date for completion has been finalized

In the first view, an end-date for release of a computer-based system has already been established and fixed

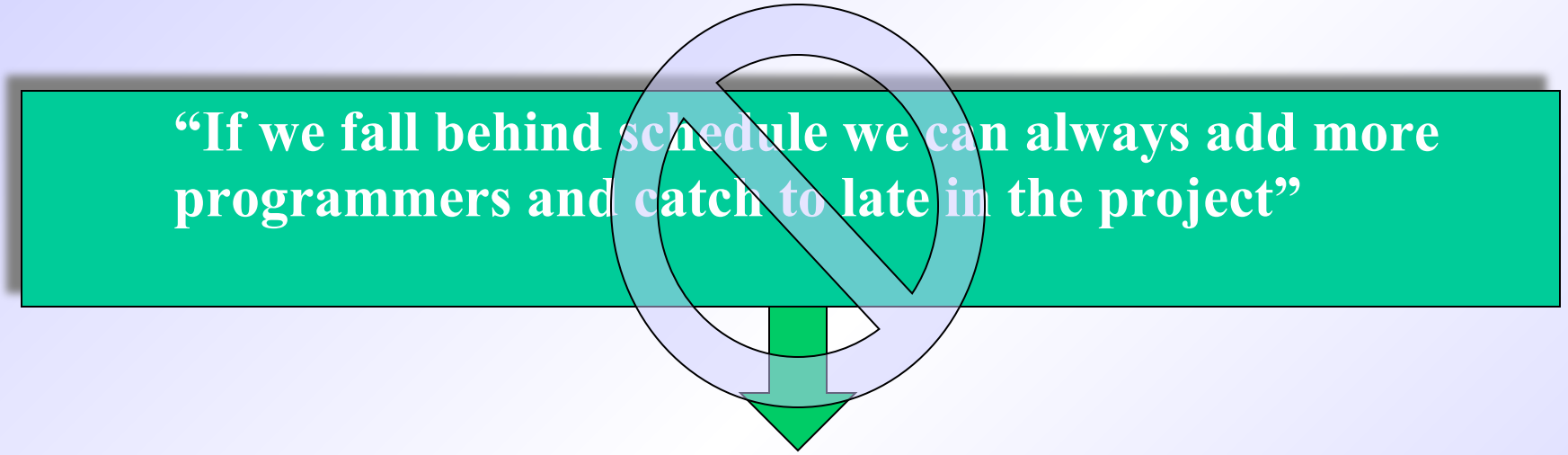
Only Rough time-frame is given

In the second view, assume that rough chronological bounds have been discussed but that the end-date is set by the software engineering organization

Basic Principles for SE Scheduling

- Compartmentalization – define distinct tasks
- Interdependency- parallel and sequential tasks
- Time allocation - assigned person days, start time, ending time)
- Effort validation - be sure resources are available
- Defined responsibilities — people must be assigned
- Defined Outcomes- each task must have an output
- Defined milestones - review for quality

People and Effort



Has a disruptive effect on the project

Schedules slip even further

People and Effort

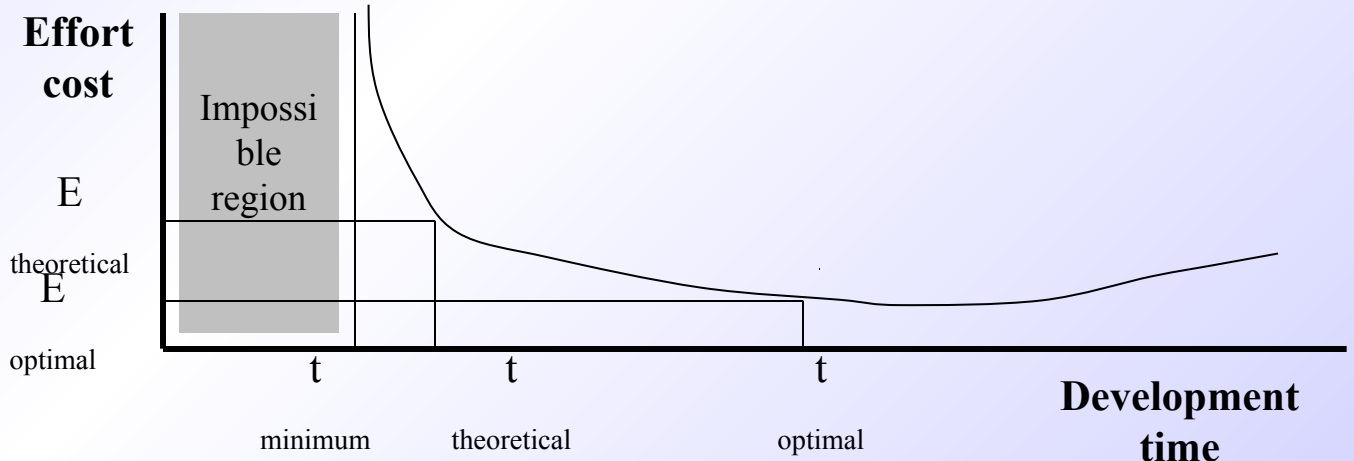
The relationship between the number of people working in software project and overall productivity is

not linear

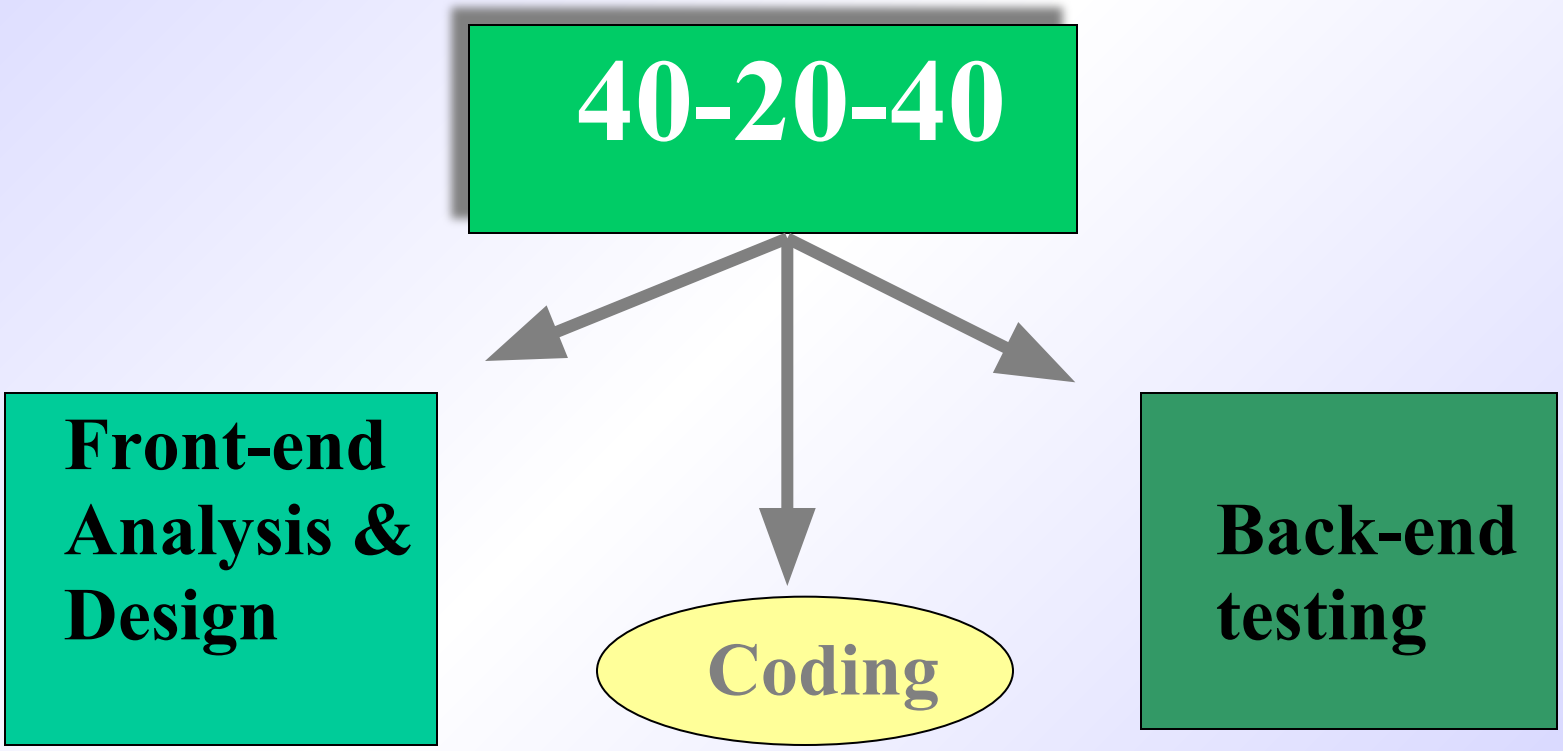
Fewer people and ***longer time period*** is a better option for software development

Effort Applied vs. Delivery Time

- There is a nonlinear relationship between effort applied and delivery time (Ref: Putnam-Norden-Rayleigh Curve)
  - Effort increases rapidly as the delivery time is reduced
- Also, delaying project delivery can reduce costs significantly as shown in the equation $\mathbf{E = L^3/(P^3t^4)}$ and in the curve below
  - E = development effort in person-months
  - L = source lines of code delivered
  - P = productivity parameter (ranging from 2000 to 12000)
  - t = project duration in calendar months

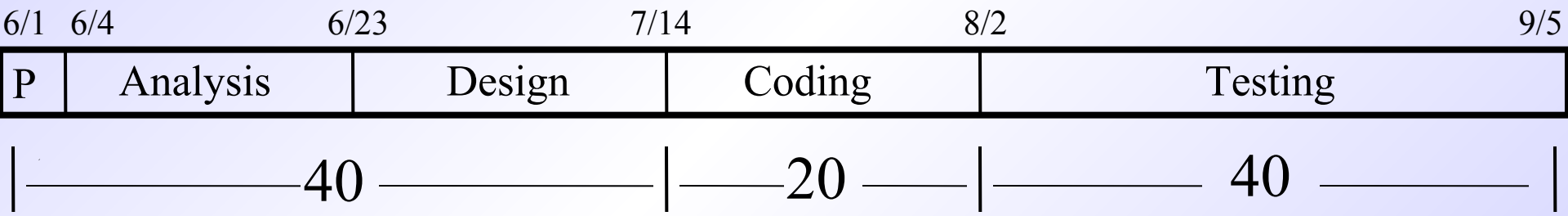


Effort Distribution

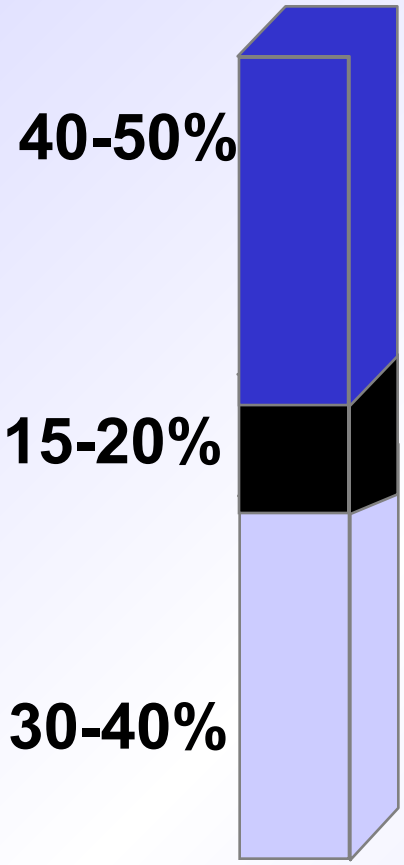


40-20-40 Distribution of Effort (continued)

Example: 100-day project

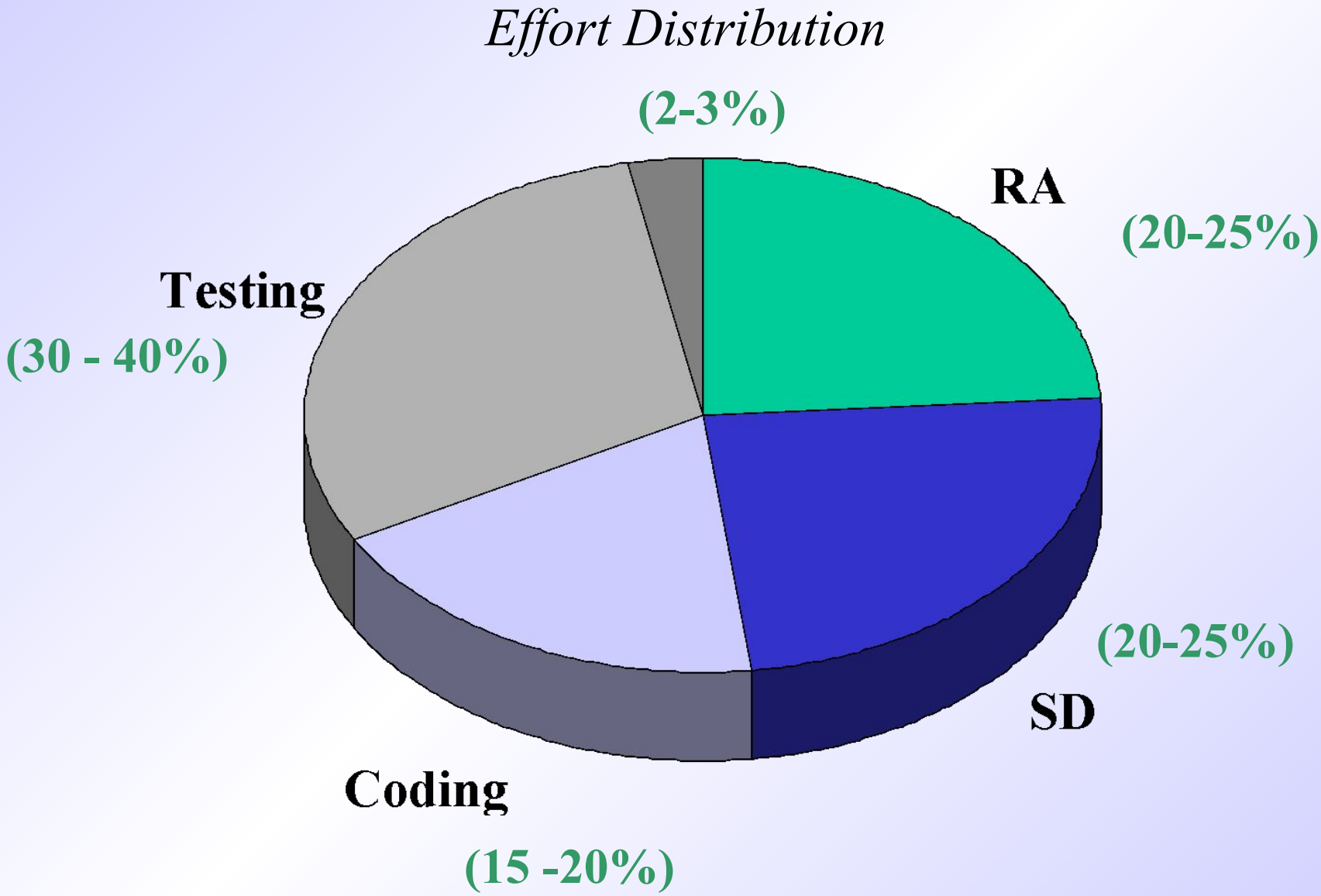


Effort Allocation



- **"front end" activities**
  - customer communication
  - analysis
  - design
  - review and modification
- **construction activities**
  - coding or code generation
- **testing and installation**
  - unit, integration
  - white-box, black box
  - regression

Effort Distribution



Task Network

Defining a Task Set

- A task set is the work breakdown structure for the project
- No single task set is appropriate for all projects and process models
  - It varies depending on the project type and the degree of rigor (based on influential factors) with which the team plans to work
- The task set should provide enough discipline to achieve high software quality
  - But it must not burden the project team with unnecessary work

Scheduling and Planning

In order to make a schedule, the following tasks must be completed:

- Identify manageable activities and tasks by decomposing the process and the product.
- Determine which tasks are dependent on the completion of others. (Which activities must occur in sequence and which can occur concurrently.)
- Allocate each task a number of work-units (often person-days), a start date and a completion date.
- Define responsibilities for the tasks (allocate them to a person or persons).
- Define outcomes of the tasks (deliverables) and milestones for the schedule.
- Review the proposed tasks, their effort allocation and start and end dates with the people involved to ensure there are no conflicts and over allocation.

Identifying Tasks

The first step :

identify the tasks required to be performed.

These tasks will comprise software engineering activities broken down for product functions.

A schedule is not a fixed entity and as such it will be refined as a project progresses.

- Initially rough

  a project schedule usually refers to the work tasks, deliverables and milestones for major software engineering activities and major product functions

- is refined in detail

  as the project progresses to refer to specific tasks and activities that must be completed for those major activities and functions.

Selecting Project Tasks

No set of tasks is appropriate for all projects.

The set of tasks that are appropriate for a project depends on a number of factors. These include:

- The process model selected. An iterative development model would require different tasks, etc... than would a waterfall model or rapid application development model.
- The type of project. A new development project has a different set of tasks to a maintenance project or to a concept development.
- The size and complexity of the product.
- The rigor required in development. This is a factor generally determined by things like product size, mission criticality, stability of requirements, etc...

Selecting Project Tasks

Many software engineering tasks have deliverables.

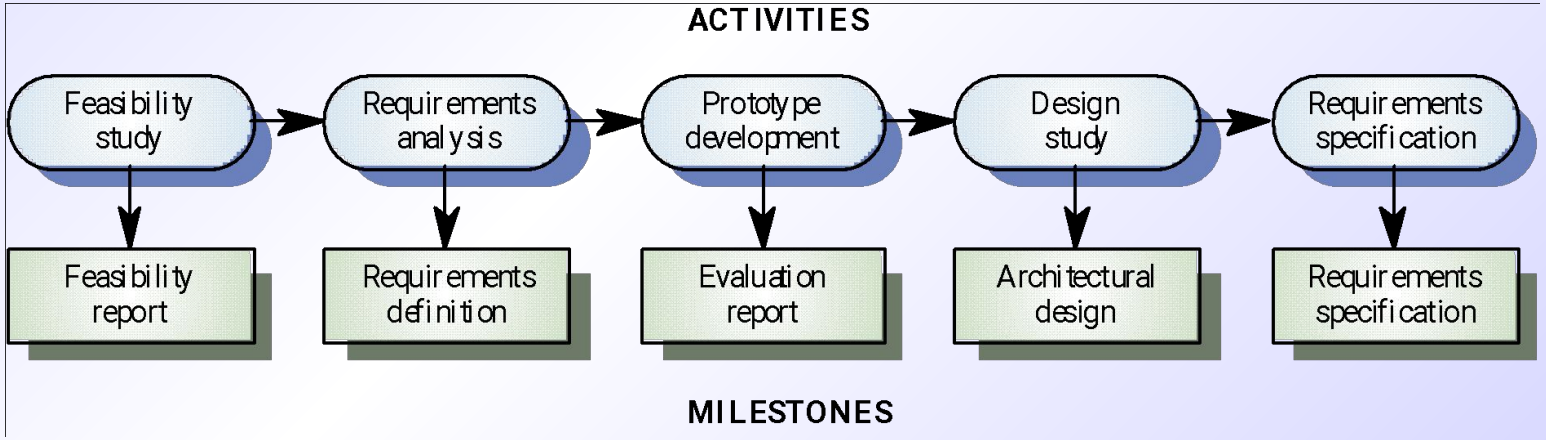
For example, the Software Requirements Specification (SRS) might be the deliverable produced as a result of requirements analysis.

Milestones are objectively identifiable points in a project. They are generally associated with the completion of a major activity. It is often a good idea to associate them with deliverables.

For example, a milestone might be established at initial requirements sign-off. This checkpoint will come after the requirements documents have been produced, inspected, corrected and signed off on. The only rule for establishing a checkpoint is that you must be able to objectively determine if that milestone has been reached.

Selecting Project Tasks

- *Milestone* = end-point of a specific, distinct software process activity or task (for each milestone a report should be presented to the management)
- *Deliverable* = project result delivered to the client
- In order to establish milestones the phases of the software process phases need be divided in basic activities/tasks.



Defining Task Sets

- determine type of project
- assess the degree of rigor required
  - identify adaptation criteria
  - compute task set selector (TSS) value
  - interpret TSS to determine degree of rigor
- select appropriate software engineering tasks

Software Project Types

- Concept Development projects
- New Application Development Projects
- Application Enhancement Project
- Application Maintenance Project
- Reengineering Project

Types of Software Projects

- Concept development projects
  - Explore some new business concept or application of some new technology
- New application development
  - Undertaken as a consequence of a specific customer request
- Application enhancement
  - Occur when existing software undergoes major modifications to function, performance, or interfaces that are observable by the end user
- Application maintenance
  - Correct, adapt, or extend existing software in ways that may not be immediately obvious to the end user
- Reengineering projects
  - Undertaken with the intent of rebuilding an existing (legacy) system in whole or in part

Degree of Rigor

Casual

- Minimum required
- Umbrella minimized
- Document requireme
- Basic prin applicable

Structured

- Framework a be applied
- Umbrella tasl applied
- Documentatic measurement be done in a s manner

Strict

- Full proces
- Degree of d high qualit
- All umbrell be applied
- Robust wor be produce

Quick Reaction

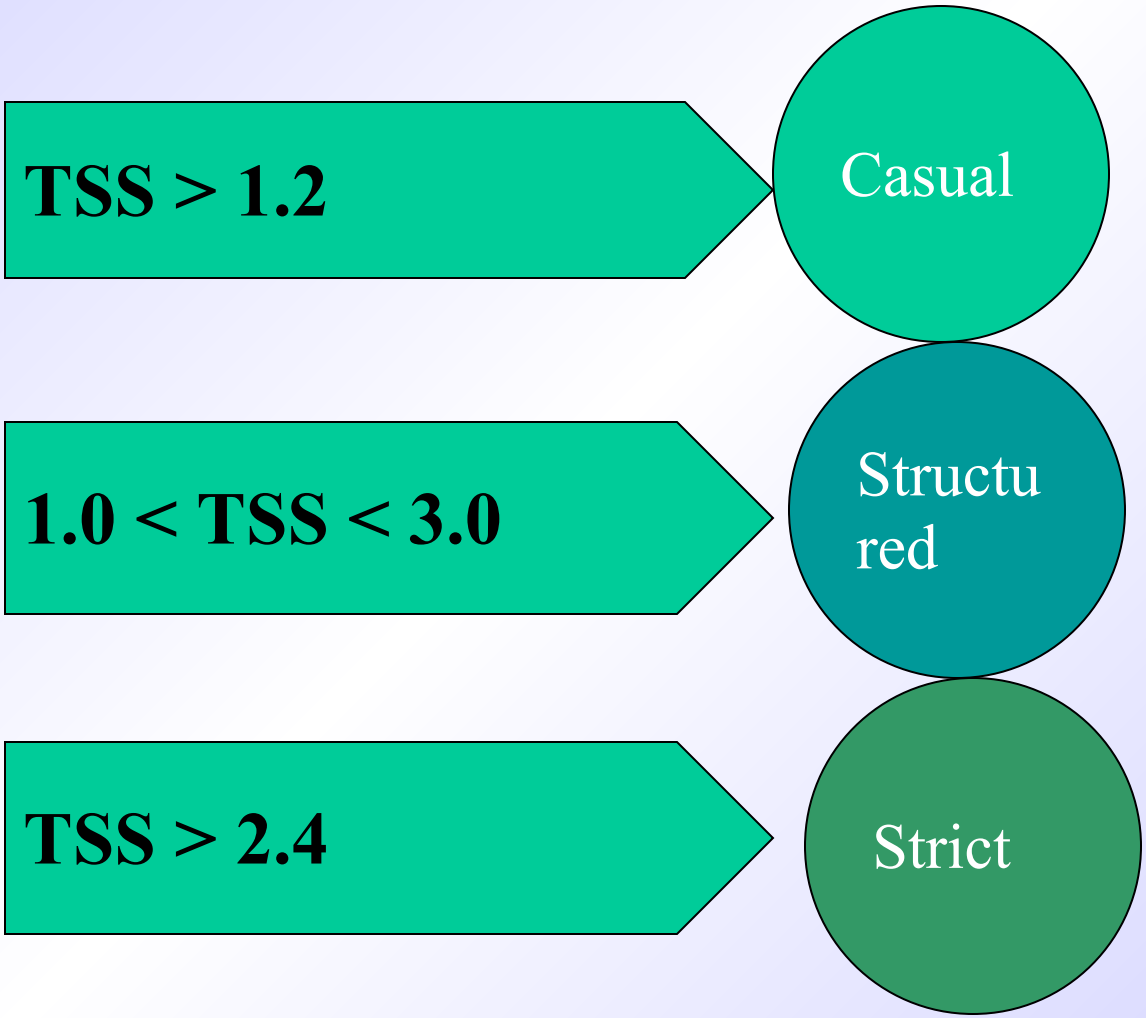
- Process framework will be applied
- Only essential tasks will be undertaken
- Documentation will be provided after product delivery

Factors that Influence a Project's Schedule

- Size of the project
- Number of potential users
- Mission criticality
- Application longevity
- Stability of requirements
- Ease of customer/developer communication
- Maturity of applicable technology
- Performance constraints
- Embedded and non-embedded characteristics
- Project staff
- Reengineering factors

Task Set selector

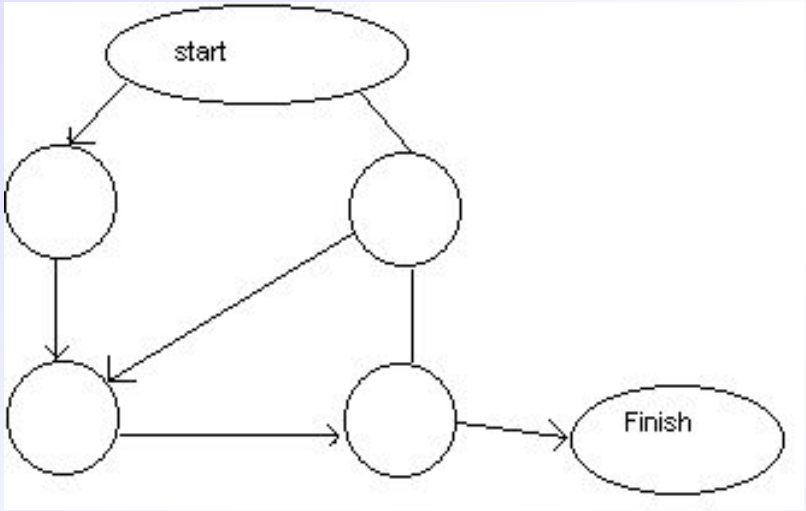
- Based on adaptation criteria, TSS is computed



Purpose of a Task Network

- Also called an activity network
- It is a graphic representation of the task flow for a project
- It depicts task length, sequence, concurrency, and dependency
- Points out inter-task dependencies to help the manager ensure continuous progress toward project completion
- The critical path
  - A single path leading from start to finish in a task network
  - It contains the sequence of tasks that must be completed on schedule if the project as a whole is to be completed on schedule
  - It also determines the minimum duration of the project

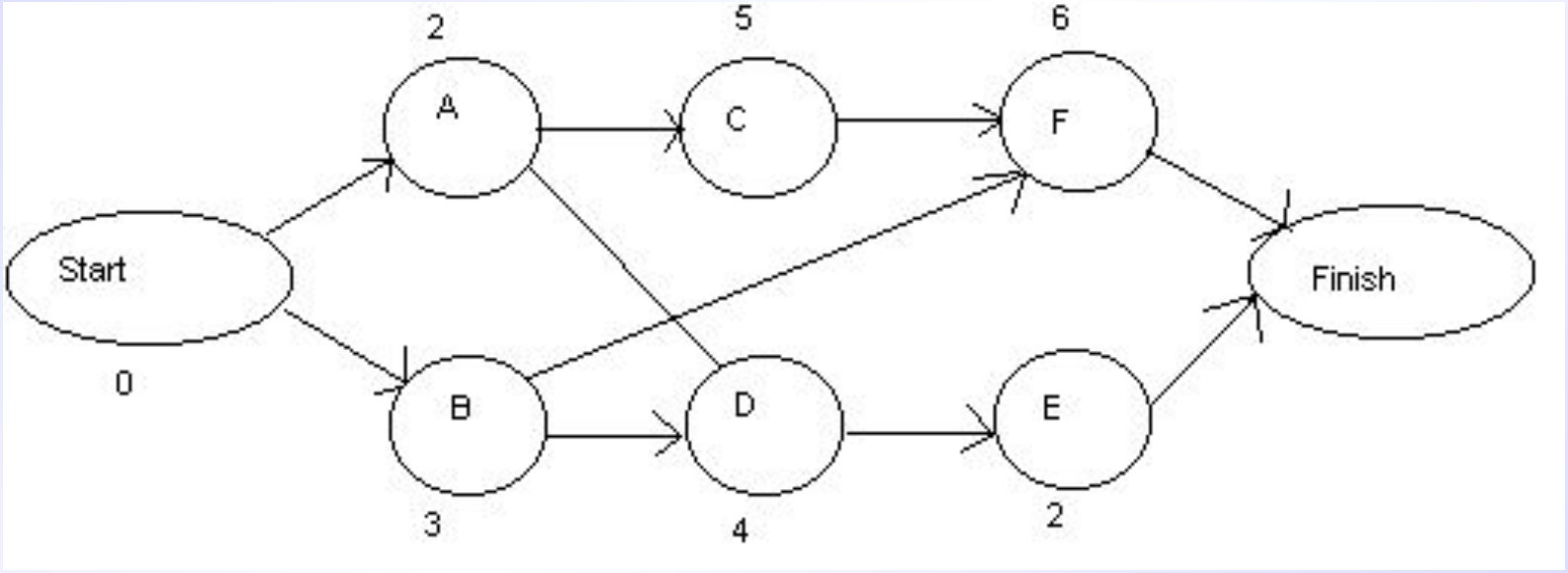
Activity Graph



Each activity has:

1. Precursor
2. Duration
3. Due date
4. End point (milestone or deliverable)

CPM
Critical Path Method



Activity	Precursor	Duration	EST	EFT	LST	LFT	Slack
Start	-	0	0	0	0	0	0
A	Start	2	0	2	0	2	0
B	Start	3	0	3	4	7	4
C	A	5	2	7	2	7	0
D	A,B	4	3	7	7	11	4
E	D	2	7	9	11	13	4
F	B,C	6	7	13	7	13	0
FINISH	E,F	0	13	13	13	13	0

EST = earliest start time, EFT = earliest finish time.

LST = latest start time, LFT = latest finish time.

Slack = (LST - EST) or (LFT - EFT).

CPM Equations

EST(START) always = 0, EFT(START) always = 0.

LST(START) always = 0, LFT(START) always = 0.

EFT(I) = EST(I) + DUR(I).

EST(I) = max(EFT of all predecessors).

LST(I) = LFT(I) - DUR(I).

LFT(I) = min(LST of all sucessors).

LFT(FINISH) = LST(FINISH) = EST(FINISH) = EFT(FINISH).

Critical path is all nodes with Slack = 0.

Task Set Refinement

1.1 **Concept scoping** determines the overall scope of the project.

is refined to

Task definition: Task 1.1 Concept Scoping

1.1.1 Identify need, benefits and potential customers;

1.1.2 Define desired output/control and input events that drive the application;

Begin Task 1.1.2

1.1.2.1 FTR: Review written description of need

FTR indicates that a formal technical review (Chapter 26) is to be conducted.

1.1.2.2 Derive a list of customer visible outputs/inputs

1.1.2.3 FTR: Review outputs/inputs with customer and revise as required;

endtask Task 1.1.2

Task Set Refinement

1.1 Concept scoping determines the overall scope of the project.

1.1.3 Define the functionality/behavior for each major function;

Begin Task 1.1.3

1.1.3.1 FTR: Review output and input data objects derived in task 1.1.2;

1.1.3.2 Derive a model of functions/behaviors;

1.1.3.3 FTR: Review functions/behaviors with customer and revise as required;

endtask Task 1.1.3

1.1.4 Isolate those elements of the technology to be implemented in software;

1.1.5 Research availability of existing software;

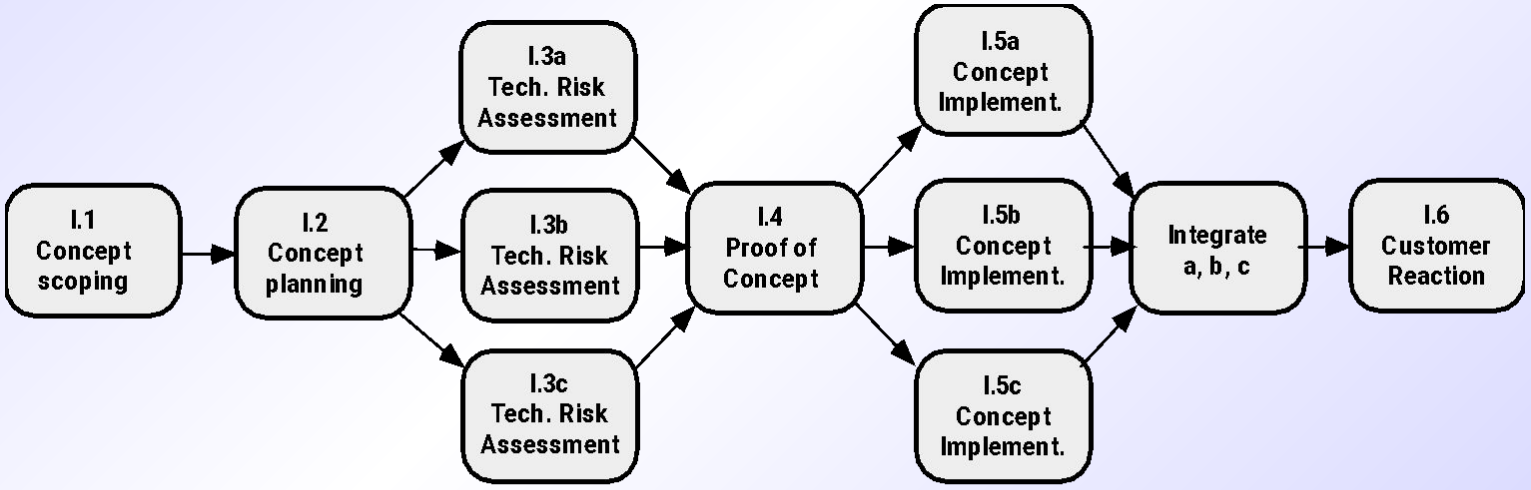
1.1.6 Define technical feasibility;

1.1.7 Make quick estimate of size;

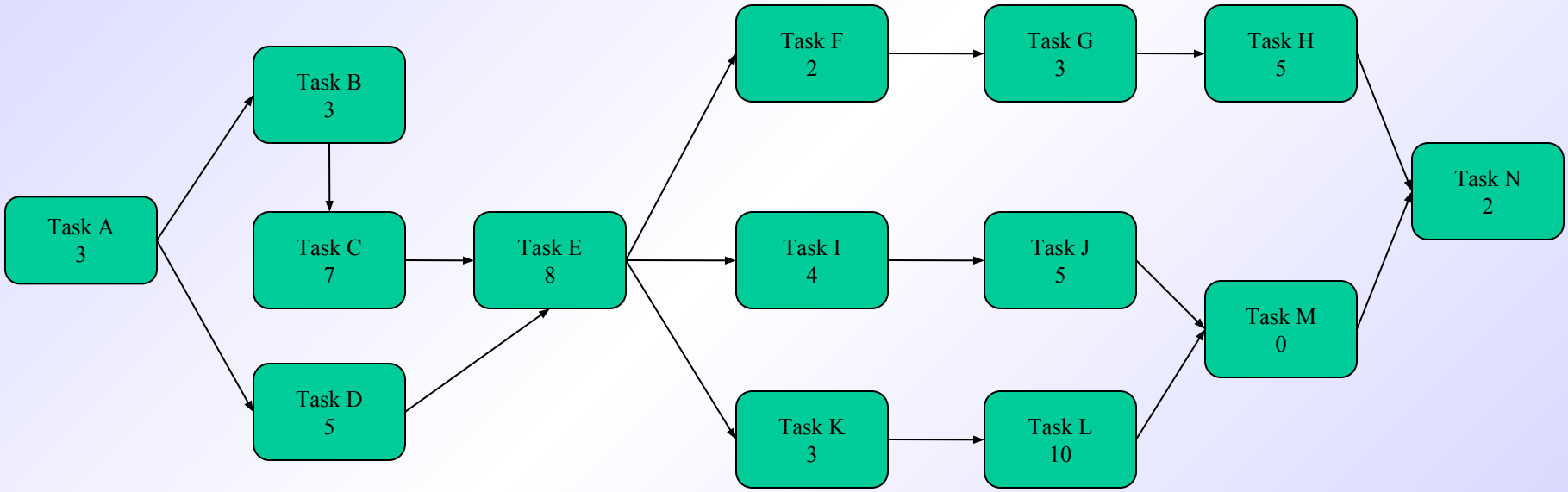
1.1.8 Create a Scope Definition;

endTask definition: Task 1.1

Define a Task Network

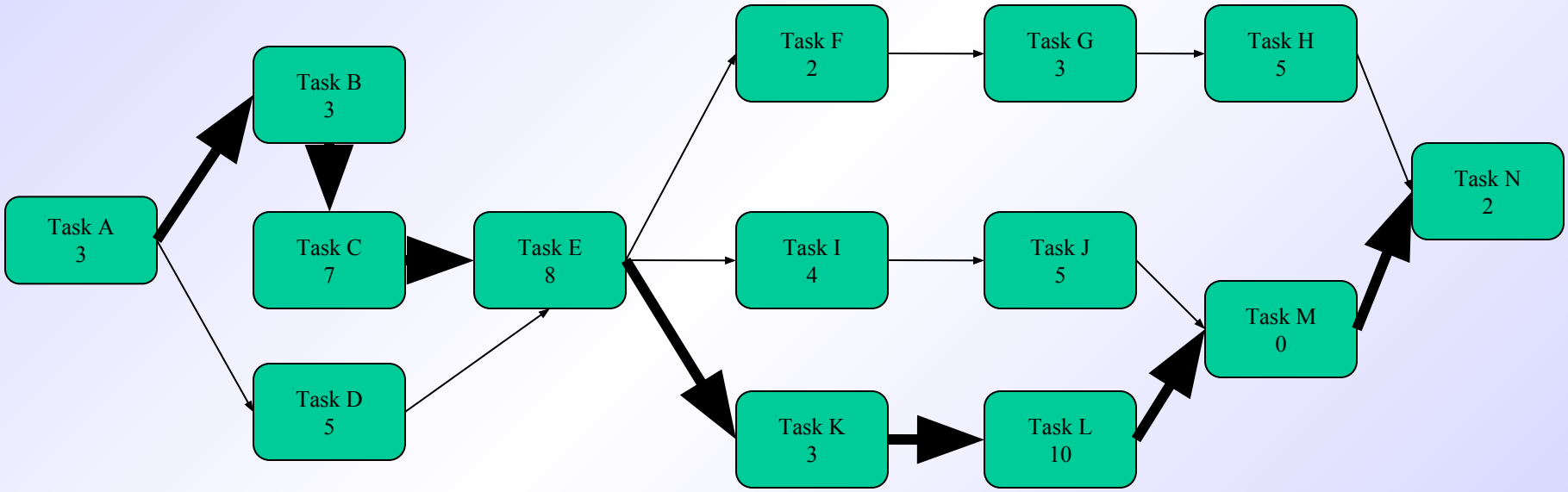


Example Task Network



Where is the critical path and what tasks are on it?

Example Task Network with Critical Path Marked



Critical path: A-B-C-E-K-L-M-N

University Questions

5. (a) Draw Gant chart and Activity Diagram for the following task set.

Activity	Dependency	Duration (in Weeks)
A1	—	1
A2	A	2
A3	—	5
A4	A1, A3	2
A5	A2, A3	3
A6	A5	7
A7	A4, A6	4

University Questions

2. (a) Draw activity diagram and Gantt chart for the following :— 10

Activity	Predecessor	Duration (in weeks)
A	—	2
B	A	3
C	—	2
D	A, C	3
E	D	4
F	A, D	3
G	B, E	2
H	G	3

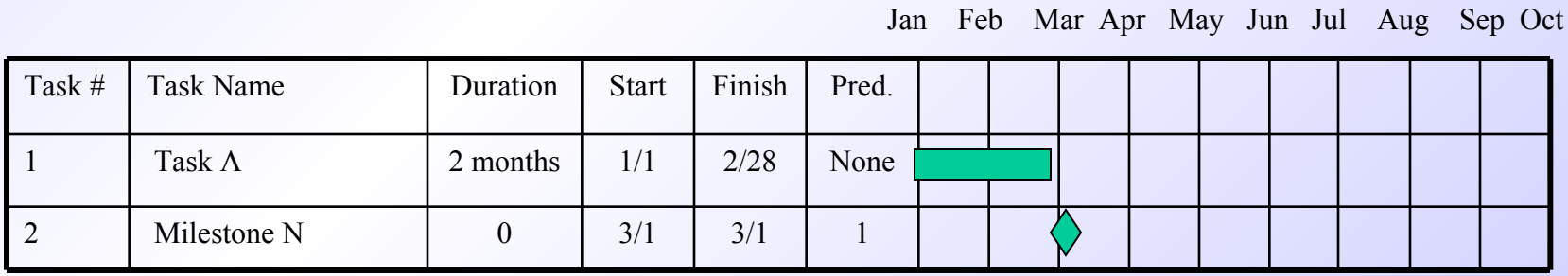
Find minimum duration required do complete the project.

(b) When does the project planning activity chart and end in a software life cycle ? Let the important activities software project managers perform during project planning. 10

Timeline Chart

Mechanics of a Timeline Chart

- Also called a Gantt chart; invented by Henry Gantt, industrial engineer, 1917
- All project tasks are listed in the far left column
- The next few columns may list the following for each task: projected start date, projected stop date, projected duration, actual start date, actual stop date, actual duration, task inter-dependencies (i.e., predecessors)
- To the far right are columns representing dates on a calendar
- The length of a horizontal bar on the calendar indicates the duration of the task
- When multiple bars occur at the same time interval on the calendar, this implies task concurrency
- A diamond in the calendar area of a specific task indicates that the task is a milestone; a milestone has a time duration of zero



CLASS EXERCISE

Timeline chart:

Task #	Task Name	Duration	Start	Finish	Pred.	4/1	4/8	4/15	4/22	4/29	5/6	5/13	5/20	5/27	6/3
A	Establish increments	3	4/1		None										
B	Analyze Inc One	3			A										
C	Design Inc One	8			B										
D	Code Inc One	7			C										
E	Test Inc One	10			D										
F	Install Inc One	5			E										
G	Analyze Inc Two	7			A, B										
H	Design Inc Two	5			G										
I	Code Inc Two	4			H										
J	Test Inc Two	6			E, I										
K	Install Inc Two	2			J										
L	Close out project	2			F, K										

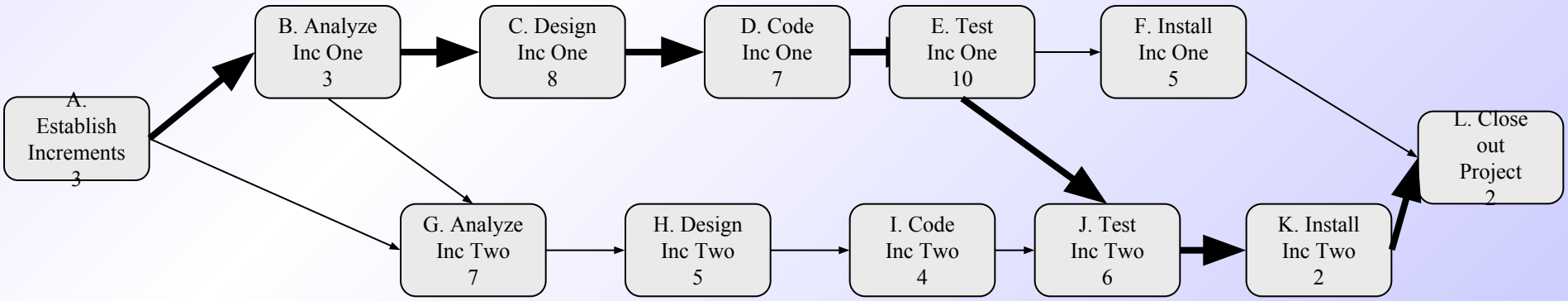
Task network and the critical path:

SOLUTION

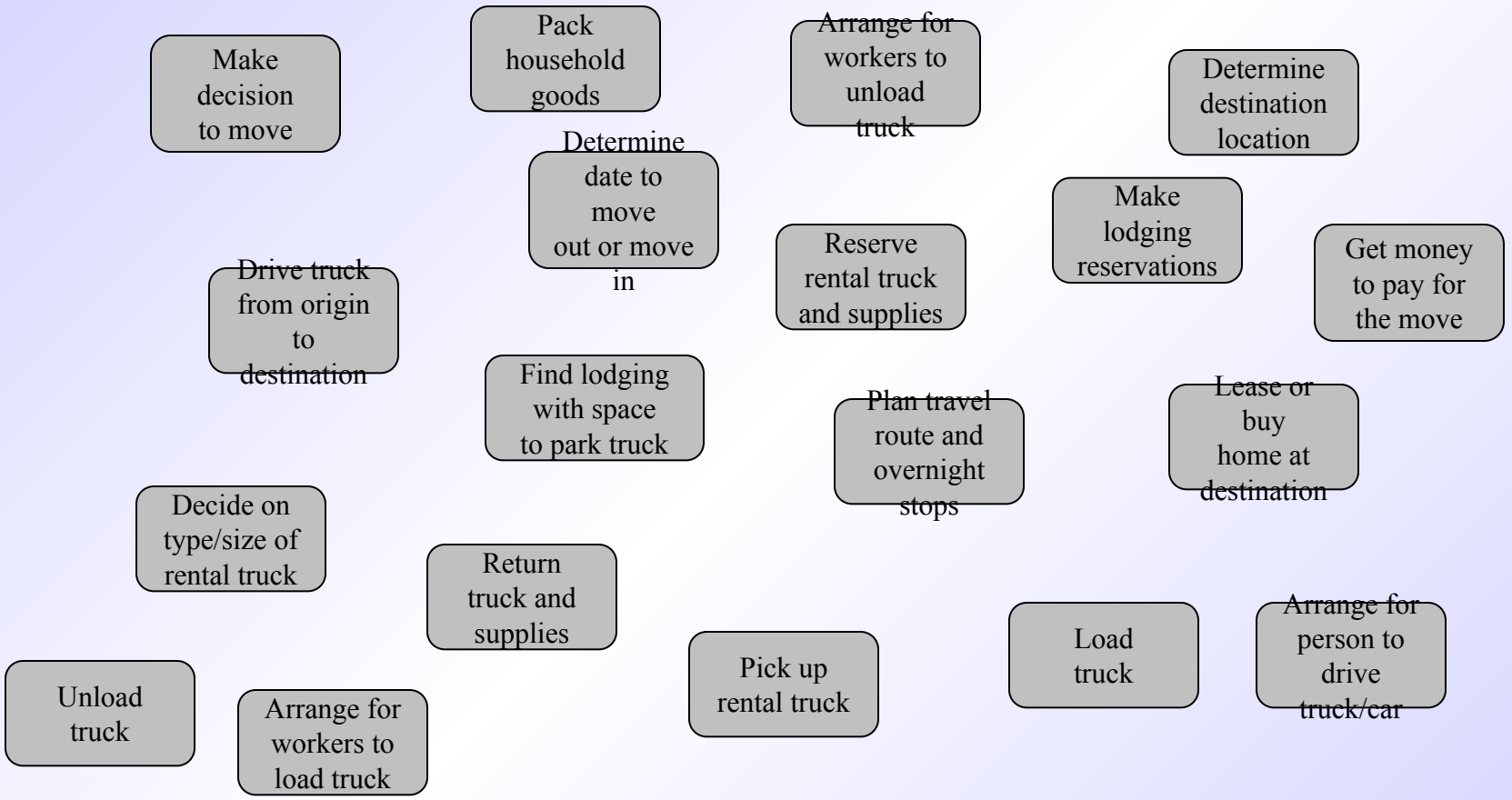
Timeline chart:

Task #	Task Name	Duration	Start	Finish	Pred.
A	Establish increments	3	4/1	4/3	None
B	Analyze Inc One	3	4/4	4/6	A
C	Design Inc One	8	4/7	4/14	B
D	Code Inc One	7	4/15	4/21	C
E	Test Inc One	10	4/22	5/1	D
F	Install Inc One	5	5/2	5/6	E
G	Analyze Inc Two	7	4/7	4/13	A, B
H	Design Inc Two	5	4/14	4/18	G
I	Code Inc Two	4	4/19	4/22	H
J	Test Inc Two	6	5/2	5/7	E, I
K	Install Inc Two	2	5/8	5/9	J
L	Close out project	2	5/10	5/11	F, K

Task network and the critical path: A-B-C-D-E-J-K-L

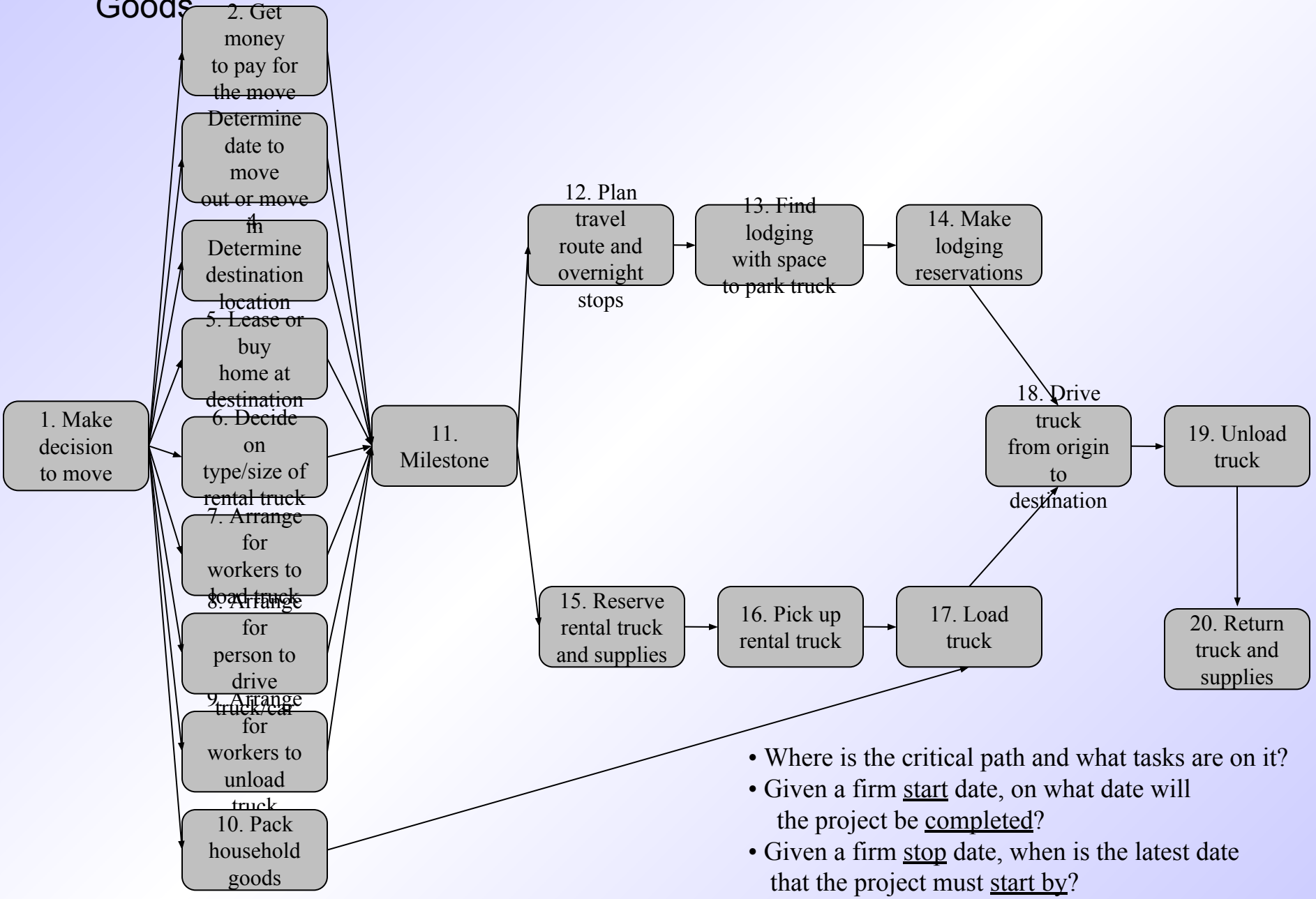


Proposed Tasks for a Long-Distance Move of 8,000 lbs of Household Goods



- Where is the critical path and what tasks are on it?
- Given a firm start date, on what date will the project be completed?
- Given a firm stop date, when is the latest date that the project must start by?

Task Network for a Long-Distance Move of 8,000 lbs of Household Goods



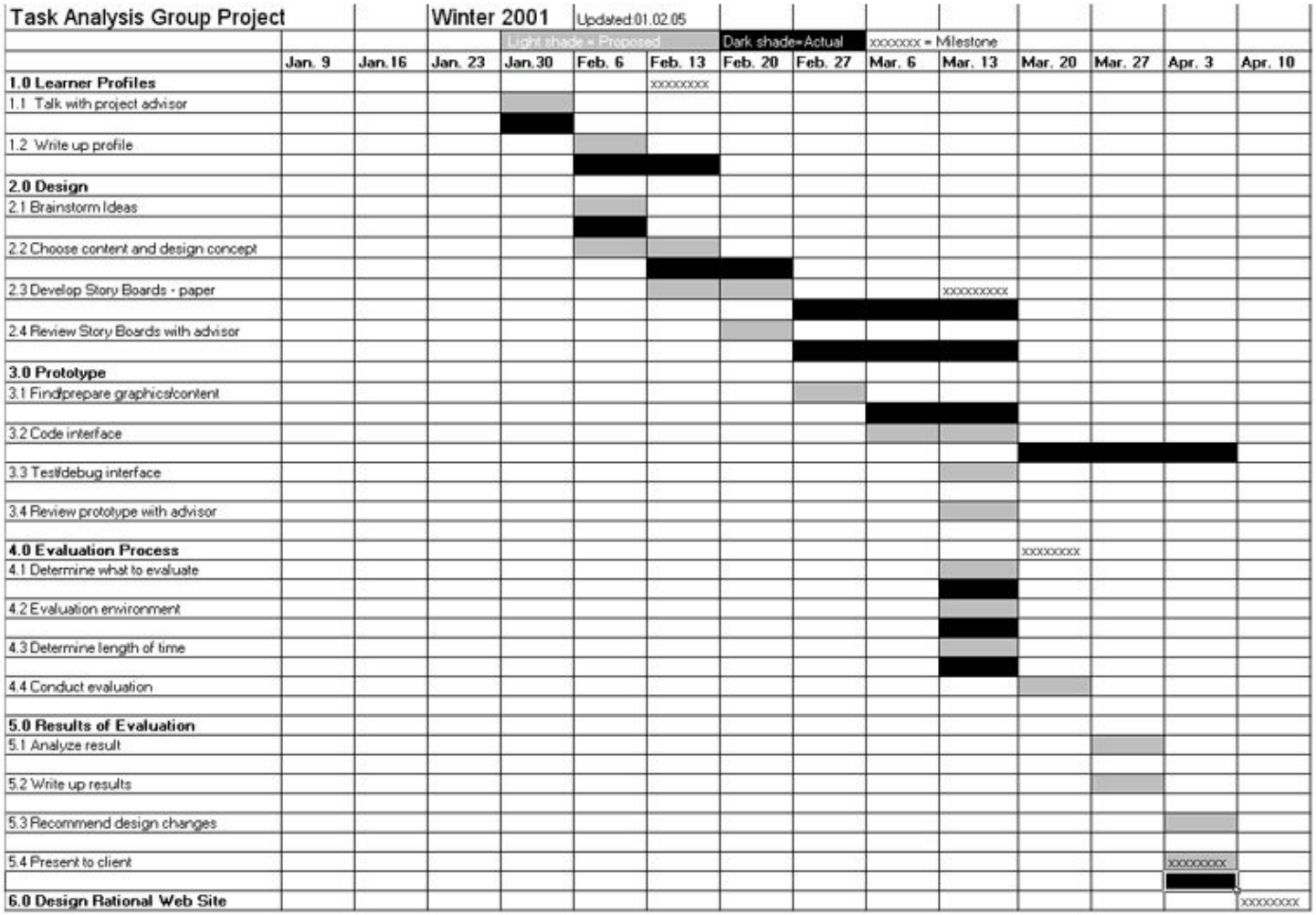
- Where is the critical path and what tasks are on it?
- Given a firm <u>start</u> date, on what date will the project be <u>completed</u>?
- Given a firm <u>stop</u> date, when is the latest date that the project must <u>start by</u>?

Timeline Chart for Long Distance Move

							Calendar		Planned time			Actual time			
Task #	Task Name	Pred.	Duration	Start Date	Stop Date	Resources									
1	Make decision to move														
2	Get money to pay for the move														
3	Determine date to move in or move out														
4	Determine destination location														
5	Lease or buy home at destination														
6	Decide on type/size of rental truck														
7	Arrange for workers to load truck														
8	Arrange for person to drive truck or car														
9	Arrange for workers to unload truck														
10	Pack household goods														
11	Milestone														
12	Plan travel route and overnight stops														
13	Find lodging with space to park truck														
14	Make lodging reservations														
15	Reserve rental truck and supplies														
16	Pick up rental truck														
17	Load truck														
18	Drive truck from origin to destination														
19	Unload truck														
20	Return truck and supplies														

Example Timeline Chart

For this particular project, the Gantt chart was useful mainly for tracking progress and visualizing how much time is left for each stage. Excel was chosen as the medium for developing the Gantt chart because all members had access to Excel and it was fairly easy to update or change without requiring any HTML coding or similar methods.



Methods for Tracking the Schedule

- Qualitative approaches
  - Conduct periodic project status meetings in which each team member reports progress and problems
  - Evaluate the results of all reviews conducted throughout the software engineering process
  - Determine whether formal project milestones (i.e., diamonds) have been accomplished by the scheduled date
  - Compare actual start date to planned start date for each project task listed in the timeline chart
  - Meet informally with the software engineering team to obtain their subjective assessment of progress to date and problems on the horizon
- Quantitative approach
  - Use earned value analysis to assess progress quantitatively

"The basic rule of software status reporting can be summarized in a single phrase: No surprises." Capers Jones

Project Control and Time Boxing

- The project manager applies control to administer project resources, cope with problems, and direct project staff
- If things are going well (i.e., schedule, budget, progress, milestones) then control should be light
- When problems occur, the project manager must apply tight control to reconcile the problems as quickly as possible. For example:
  - Staff may be redeployed
  - The project schedule may be redefined
- Severe deadline pressure may require the use of time boxing
  - An incremental software process is applied to the project
  - The tasks associated with each increment are "time-boxed" (i.e., given a specific start and stop time) by working backward from the delivery date
  - The project is not allowed to get "stuck" on a task
  - When the work on a task hits the stop time of its box, then work ceases on that task and the next task begins
  - This approach succeeds based on the premise that when the time-box boundary is encountered, it is likely that 90% of the work is complete
  - The remaining 10% of the work can be
    - Delayed until the next increment
    - Completed later if required

Milestones for OO Projects

- Task parallelism in object-oriented projects makes project tracking more difficult to do than non-OO projects because a number of different activities can be happening at once
- Sample milestones
  - Object-oriented analysis completed
  - Object-oriented design completed
  - Object-oriented coding completed
  - Object-oriented testing completed
- Because the object-oriented process is an iterative process, each of these milestones may be revisited as different increments are delivered to the customer

Earned Value Analysis

Description of Earned Value Analysis

- Earned value analysis is a measure of progress by assessing the percent of completeness for a project
- It gives accurate and reliable readings of performance very early into a project
- It provides a common value scale (i.e., time) for every project task, regardless of the type of work being performed
- The total hours to do the whole project are estimated, and every task is given an earned value based on its estimated percentage of the total

Determining Earned Value

- Compute the budgeted cost of work scheduled (BCWS) for each work task i in the schedule
  - The BCWS is the effort planned; work is estimated in person-hours or person-days for each task
  - To determine progress at a given point along the project schedule, the value of BCWS is the sum of the $BCWS_i$ values of all the work tasks that should have been completed by that point of time in the project schedule
- Sum up the BCWS values for all work tasks to derive the budget at completion (BAC)
- Compute the value for the budgeted cost of work performed (BCWP)
  - BCWP is the sum of the BCWS values for all work tasks that have actually been completed by a point of time on the project schedule

Progress Indicators provided through Earned Value Analysis

- SPI = BCWP/BCWS
  - Schedule performance index (SPI) is an indication of the efficiency with which the project is utilizing scheduled resources
  - SPI close to 1.0 indicates efficient execution of the project schedule
- SV = BCWP – BCWS
  - Schedule variance (SV) is an absolute indication of variance from the planned schedule
- PSFC = BCWS/BAC
  - Percent scheduled for completion (PSFC) provides an indication of the percentage of work that should have been completed by time t
- PC = BCWP/BAC
  - Percent complete (PC) provides a quantitative indication of the percent of work that has been completed at a given point in time t
- ACWP = sum of BCWP as of time t
  - Actual cost of work performed (ASWP) includes all tasks that have been completed by a point in time t on the project schedule
- CPI = BCWP/ACWP
  - A cost performance index (CPI) close to 1.0 provides a strong indication that the project is within its defined budget
- CV = BCWP – ACWP
  - The cost variance is an absolute indication of cost savings (against planned costs) or shortfall at a particular stage of a project