Software Engineering

Lecture-25

Software Project Scheduling

Topics Covered

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

INTRODUCTION

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

Quote from Napoleon

"Any commander-in-chief who undertakes to carry out a plan which he considers defective is at fault; he must put forth his reasons, insist on the plan being changed, and finally tender his resignation rather than be the instrument of his army's downfall."

Handling Unrealistic Deadlines

- Perform a detailed estimate using historical data from past projects; determine the estimated effort and duration for the project
- Using an incremental model, develop a software engineering strategy that will deliver critical functionality by the imposed deadline, but delay other functionality until later; document the plan
- Meet with the customer and (using the detailed estimate) explain why the imposed deadline is unrealistic
  - Be certain to note that all estimates are based on performance on past projects
  - Also be certain to indicate the percent improvement that would be required to achieve the deadline as it currently exists

1) Offer the incremental development strategy as an alternative and offer some options
   - Increase the budget and bring on additional resources to try to finish sooner
   - Remove many of the software functions and capabilities that were requested
   - Dispense with reality and wish the project complete using the prescribed schedule; then point out that project history and your estimates show that this is unrealistic and will result in a disaster

Project Scheduling

General Practices

- On large projects, hundreds of small tasks must occur to accomplish a larger goal
  - Some of these tasks lie outside the mainstream and may be completed without worry of impacting on the project completion date
  - Other tasks lie on the critical path; if these tasks fall behind schedule, the completion date of the entire project is put into jeopardy
- Project manager's objectives
  - Define all project tasks
  - Build an activity network that depicts their interdependencies
  - Identify the tasks that are critical within the activity network
  - Build a timeline depicting the planned and actual progress of each task
  - Track task progress to ensure that delay is recognized "one day at a time"
  - To do this, the schedule should allow progress to be monitored and the project to be controlled

(More on next slide)

GENERAL PRACTICES (CONTINUED)

- Software project scheduling distributes estimated effort across the planned project duration by allocating the effort to specific tasks
- During early stages of project planning, a macroscopic schedule is developed identifying all major process framework activities and the product functions to which they apply
- Later, each task is refined into a detailed schedule where specific software tasks are identified and scheduled
- Scheduling for projects can be viewed from two different perspectives
  - In the first view, an end-date for release of a computer-based system has already been established and fixed
    - The software organization is constrained to distribute effort within the prescribed time frame
  - In the second view, assume that rough chronological bounds have been discussed but that the end-date is set by the software engineering organization
    - Effort is distributed to make best use of resources and an end-date is defined after careful analysis of the software
  - The first view is encountered far more often that the second

BASIC PRINCIPLES FOR PROJECT SCHEDULING

- Compartmentalization
  - The project must be compartmentalized into a number of manageable activities, actions, and tasks; both the product and the process are decomposed
- Interdependency
  - The interdependency of each compartmentalized activity, action, or task must be determined
  - Some tasks must occur in sequence while others can occur in parallel
  - Some actions or activities cannot commence until the work product produced by another is available
- Time allocation
  - Each task to be scheduled must be allocated some number of work units
  - In addition, each task must be assigned a start date and a completion date that are a function of the interdependencies
  - Start and stop dates are also established based on whether work will be conducted on a full-time or part-time basis

(More on next slide)

BASIC PRINCIPLES FOR PROJECT SCHEDULING (CONTINUED)

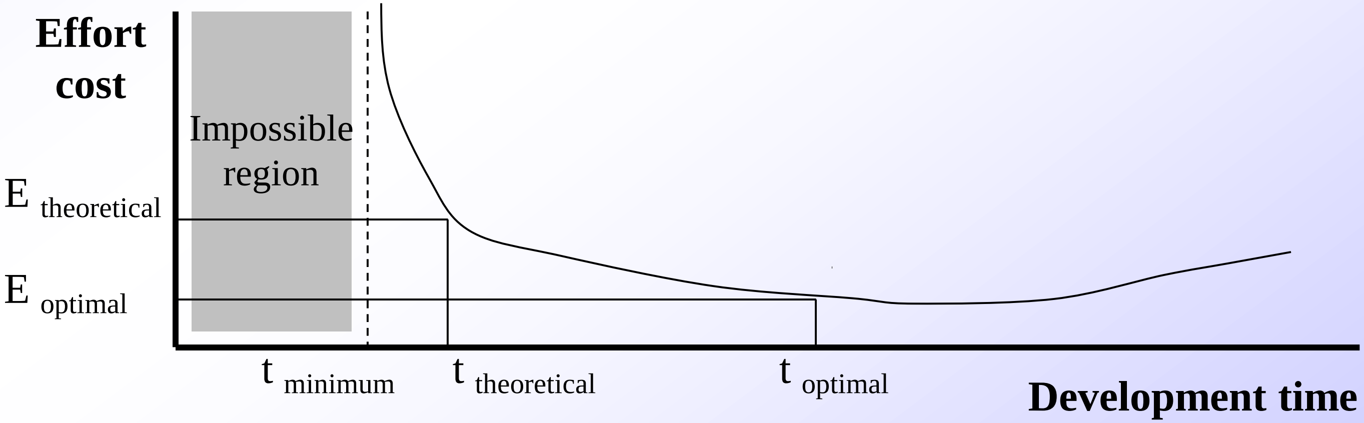
- Effort validation
  - Every project has a defined number of people on the team
  - As time allocation occurs, the project manager must ensure that no more than the allocated number of people have been scheduled at any given time
- Defined responsibilities
  - Every task that is scheduled should be assigned to a specific team member
- Defined outcomes
  - Every task that is scheduled should have a defined outcome for software projects such as a work product or part of a work product
  - Work products are often combined in deliverables
- Defined milestones
  - Every task or group of tasks should be associated with a project milestone
  - A milestone is accomplished when one or more work products has been reviewed for quality and has been approved

Relationship Between People and Effort

- Common management myth: *If we fall behind schedule, we can always add more programmers and catch up later in the project*
  - This practice actually has a disruptive effect and causes the schedule to slip even further
  - The added people must learn the system
  - The people who teach them are the same people who were earlier doing the work
  - During teaching, no work is being accomplished
  - Lines of communication (and the inherent delays) increase for each new person added

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 **$E = L^3/(P^3t^4)$** 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



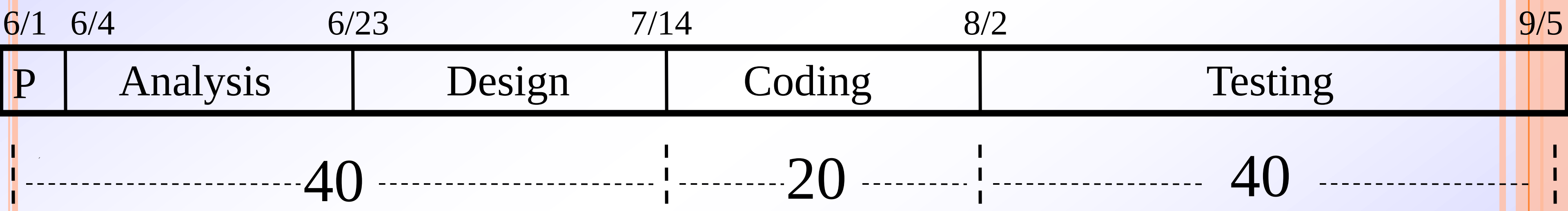
40-20-40 Distribution of Effort

- A recommended distribution of effort across the software process is 40% (analysis and design), 20% (coding), and 40% (testing)
- Work expended on project planning rarely accounts for more than 2 - 3% of the total effort
- Requirements analysis may comprise 10 - 25%
  - Effort spent on prototyping and project complexity may increase this
- Software design normally needs 20 – 25%
- Coding should need only 15 - 20% based on the effort applied to software design
- Testing and subsequent debugging can account for 30 - 40%
  - Safety or security-related software requires more time for testing

(More on next slide)

40-20-40 DISTRIBUTION OF EFFORT (CONTINUED)

Example: 100-day project



6/1 6/4 6/23 7/14 8/2 9/5

P | Analysis | Design | Coding | Testing

40 20 40

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

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

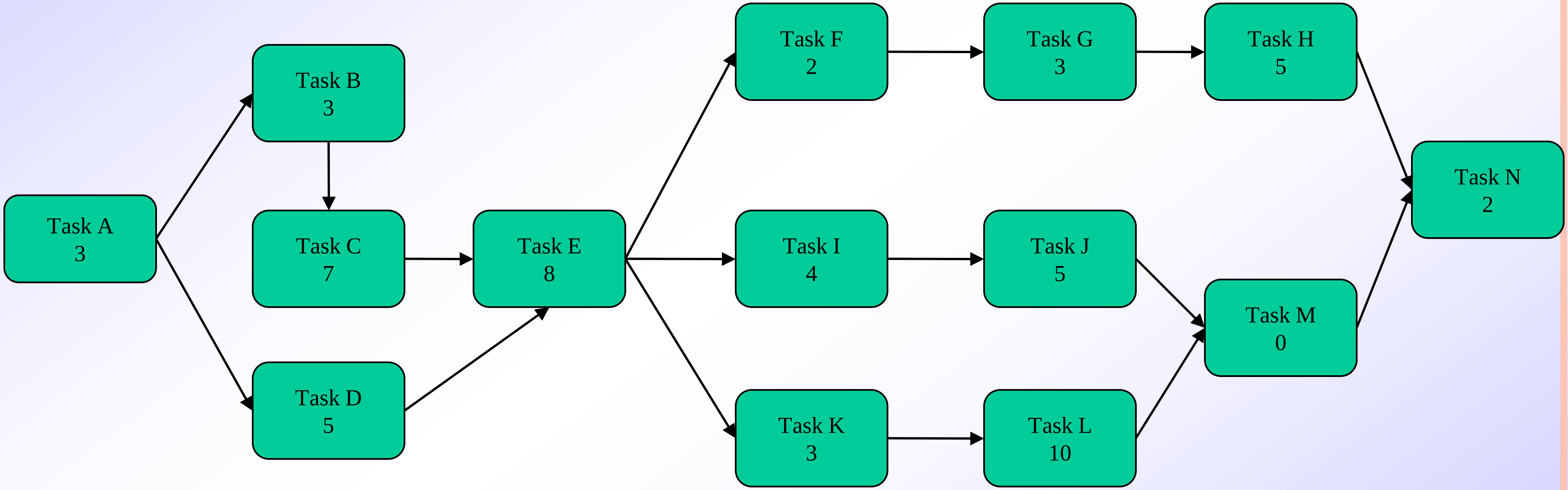
Factors that Influence a Project’s Schedule

- 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

Purpose of a Task Network

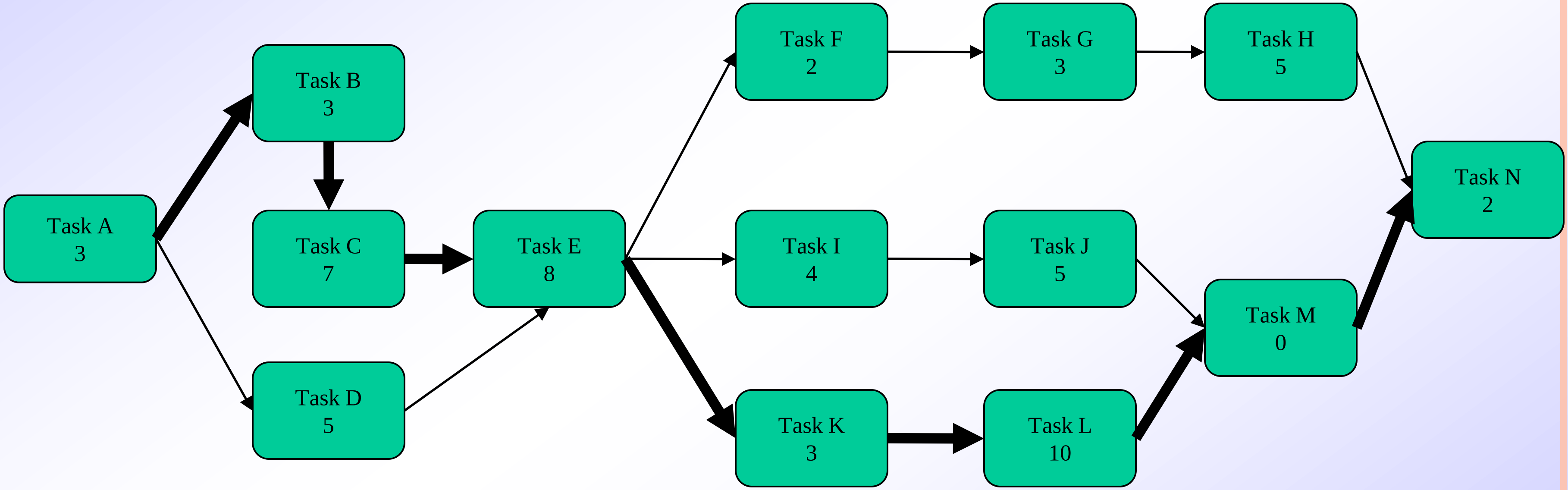
- Also called an activity network
- It is a graphic representation of the task flow for a project
- It depicts task length, sequence, concurrency, and dependency
- Points out inter-task dependencies to help the manager ensure continuous progress toward project completion
- The critical path
  - A single 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

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

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

Jan Feb Mar Apr May Jun Jul Aug Sep Oct

Task #	Task Name	Duration	Start	Finish	Pred.									
1	Task A	2 months	1/1	2/28	None	🟩	🟩							
2	Milestone N	0	3/1	3/1	1			◆						

CLASS EXERCISE

Timeline chart:

4/1 4/8 4/15 4/22 4/29 5/6 5/13 5/20 5/27 6/3

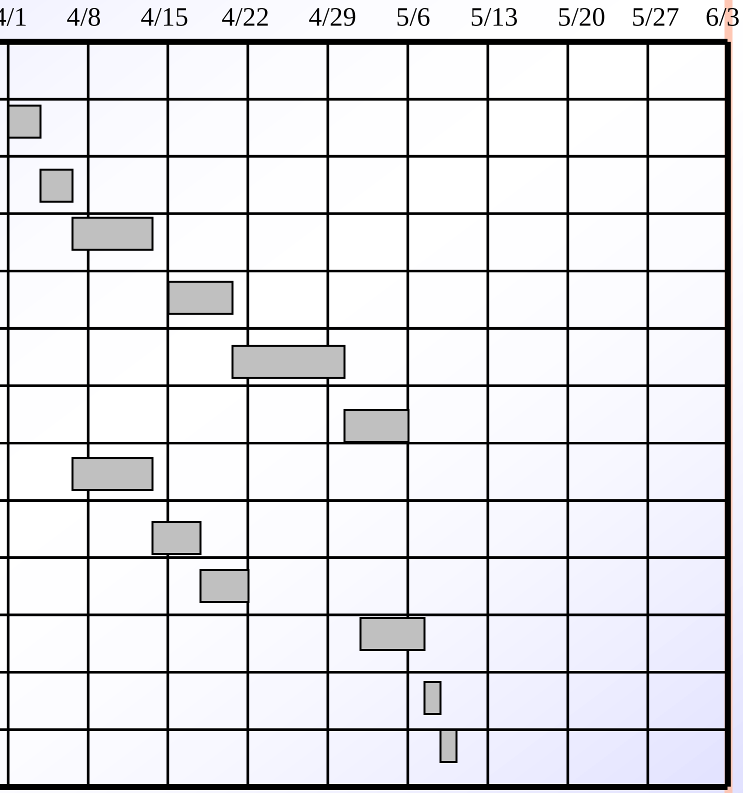
Task #	Task Name	Duration	Start	Finish	Pred.									
A	Establish increments	3	4/1		None									
B	Analyze Inc One	3			A									
C	Design Inc One	8			B									
D	Code Inc One	7			C									
E	Test Inc One	10			D									
F	Install Inc One	5			E									
G	Analyze Inc Two	7			A, B									
H	Design 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:

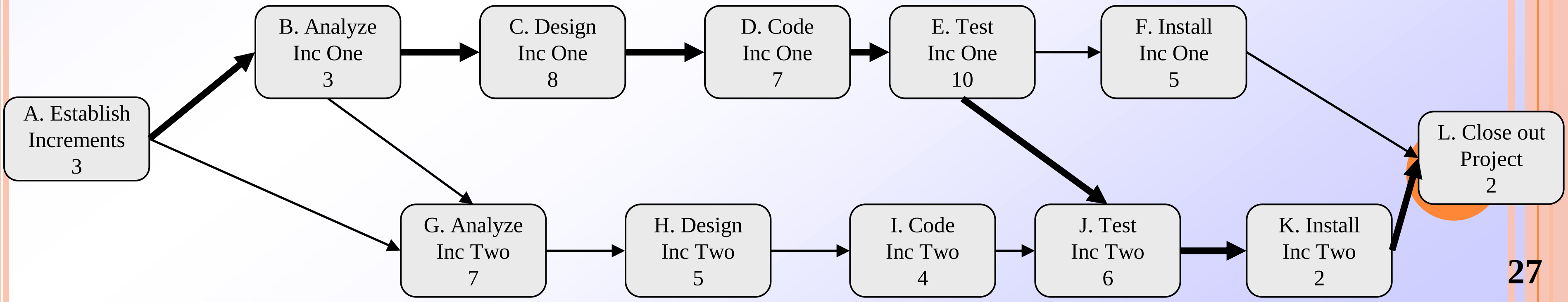
SOLUTION

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



A. Establish Increments 3

B. Analyze Inc One 3

C. Design Inc One 8

D. Code Inc One 7

E. Test Inc One 10

F. Install Inc One 5

G. Analyze Inc Two 7

H. Design Inc Two 5

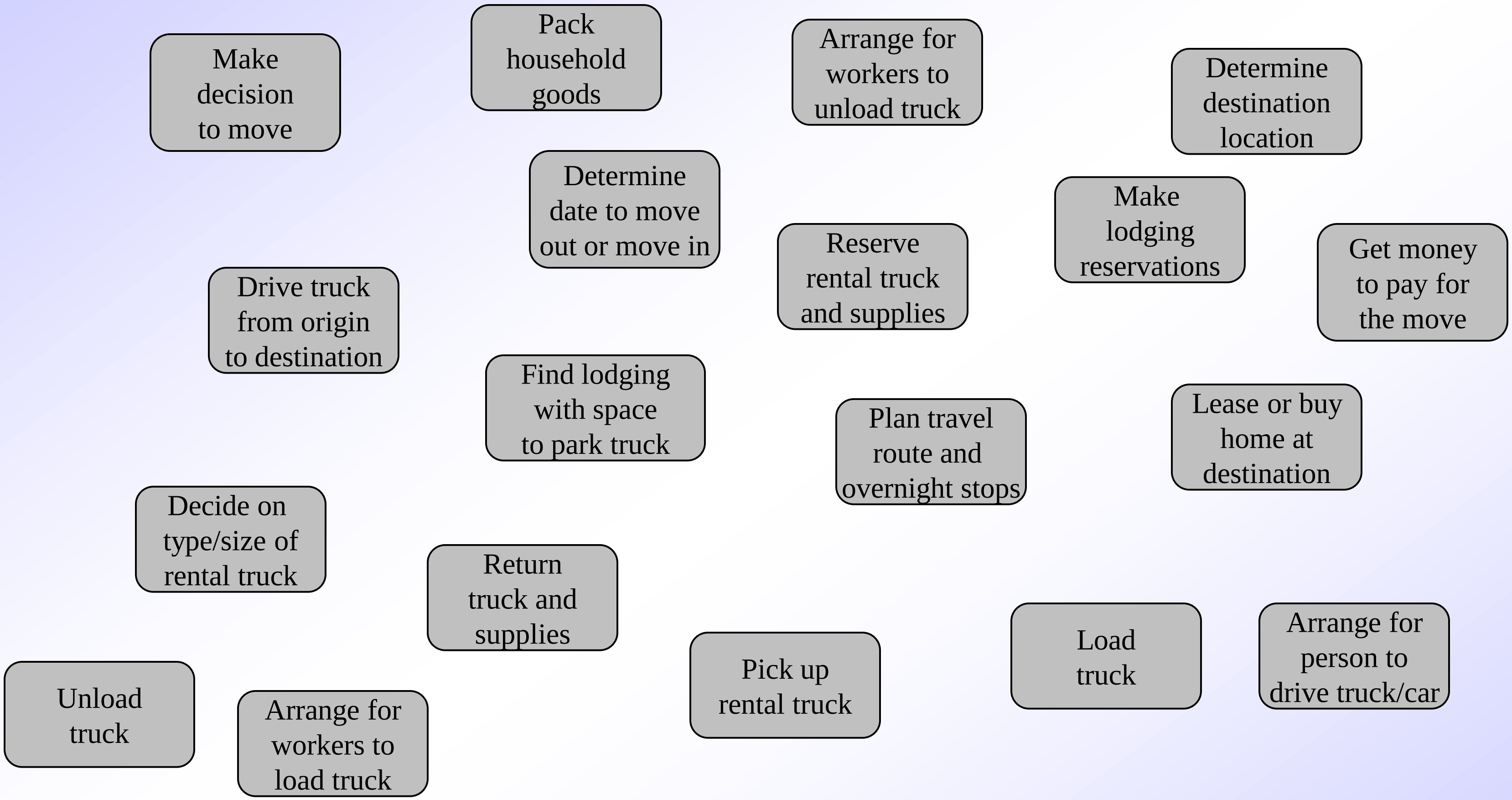
I. Code Inc Two 4

J. Test Inc Two 6

K. Install Inc Two 2

L. Close out Project 2

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



Pack household goods

Make decision to move

Arrange for workers to unload truck

Determine destination location

Determine date to move out or move in

Make lodging reservations

Reserve rental truck and supplies

Get money to pay for the move

Drive truck from origin to destination

Find lodging with space to park truck

Lease or buy home at destination

Plan travel route and overnight stops

Decide on type/size of rental truck

Return truck and supplies

Load truck

Arrange for person to drive truck/car

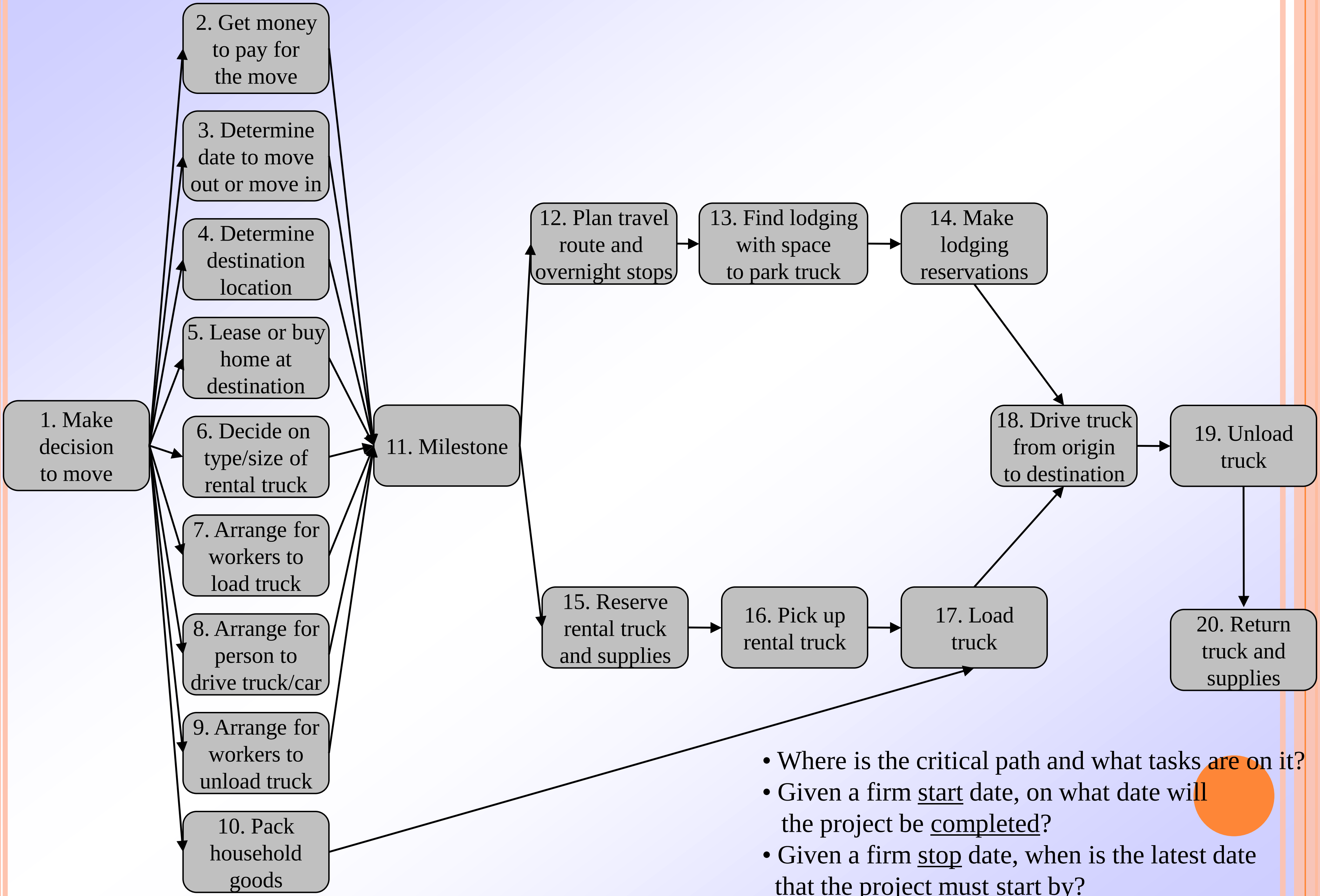
Pick up rental truck

Unload truck

Arrange for workers to load truck

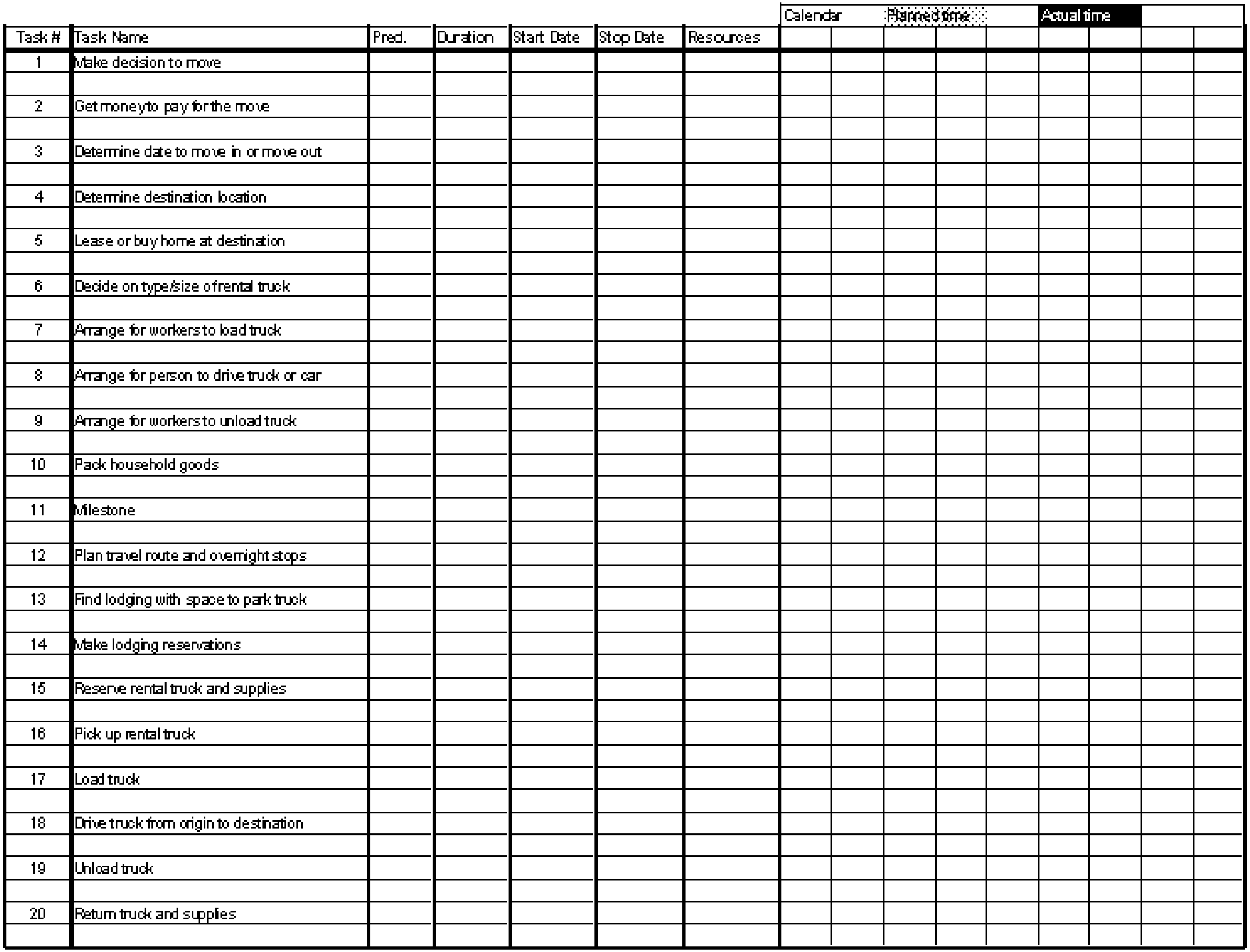
- Where is the critical path and what tasks are on it?
- Given a firm start date, on what date will the project be completed?
- Given a firm stop date, when is the latest date that the project must start by?

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


1. Make decision to move
2. Get money to pay for the move
3. Determine date to move out or move in
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/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

- 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?

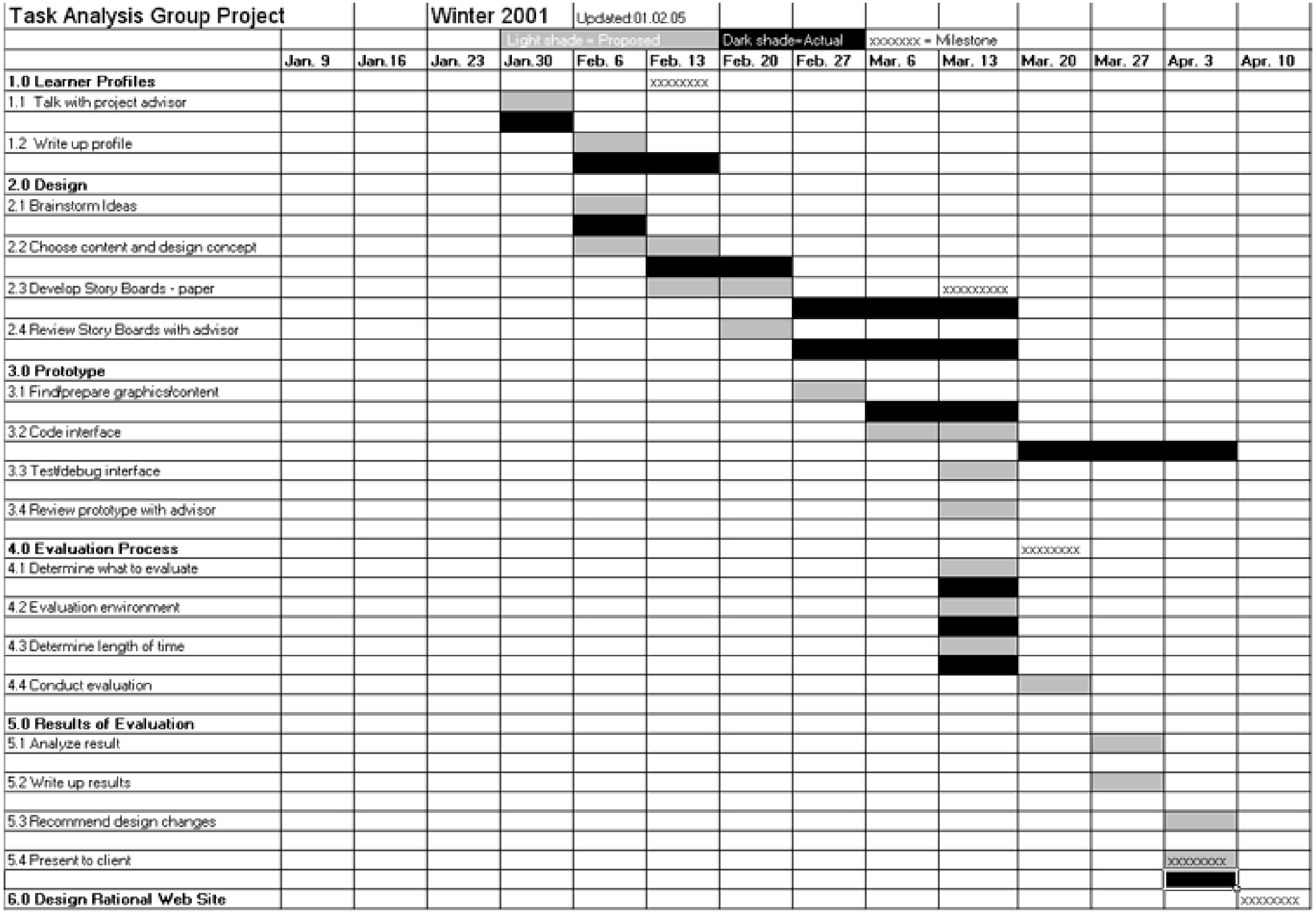
TIMELINE CHART FOR LONG DISTANCE MOVE



							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.



Task Analysis Group Project			Winter 2001		Updated:01.02.05									
				Light shade = Proposed			Dark shade=Actual		xxxxxxx = Milestone					
	Jan. 9	Jan.16	Jan. 23	Jan.30	Feb. 6	Feb. 13	Feb. 20	Feb. 27	Mar. 6	Mar. 13	Mar. 20	Mar. 27	Apr. 3	Apr. 10
1.0 Learner Profiles						xxxxxxxx								
1.1 Talk with project advisor				░										
				█										
1.2 Write up profile					░									
					█	█								
2.0 Design														
2.1 Brainstorm Ideas					░									
					█									
2.2 Choose content and design concept					░	░								
						█	█							
2.3 Develop Story Boards - paper						░	░			xxxxxxxxx				
								█	█	█				
2.4 Review Story Boards with advisor							░							
								█	█	█				
3.0 Prototype														
3.1 Find/prepare graphics/content								░						
									█	█				
3.2 Code interface									░	░				
											█	█	█	
3.3 Test/debug interface										░				
3.4 Review prototype with advisor										░				
4.0 Evaluation Process											xxxxxxxx			
4.1 Determine what to evaluate										░				
										█				
4.2 Evaluation environment										░				
										█				
4.3 Determine length of time										░				
										█				
4.4 Conduct evaluation											░			
5.0 Results of Evaluation														
5.1 Analyze result												░		
5.2 Write up results												░		
5.3 Recommend design changes													░	
5.4 Present to client													░ xxxxxxxx	
													█	
6.0 Design Rational Web Site														xxxxxxxx

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

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

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