Software Project Scheduling

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

Software Project Planning

- Software project planning encompasses five major activities
  - Estimation, scheduling, risk analysis, quality management planning, and change management planning
- Estimation determines how much money, effort, resources, and time it will take to build a specific system or product
- The software team first estimates
  - The work to be done
  - The resources required
  - The time that will elapse from start to finish
- Then they establish a project schedule that
  - Defines tasks and milestones
  - Identifies who is responsible for conducting each task
  - Specifies the inter-task dependencies

Why Are Projects Late?

- 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

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

"Overly optimistic scheduling doesn't result in shorter actual schedules, it results in longer ones"

-Steve McConnell

Handling Unrealistic Deadlines

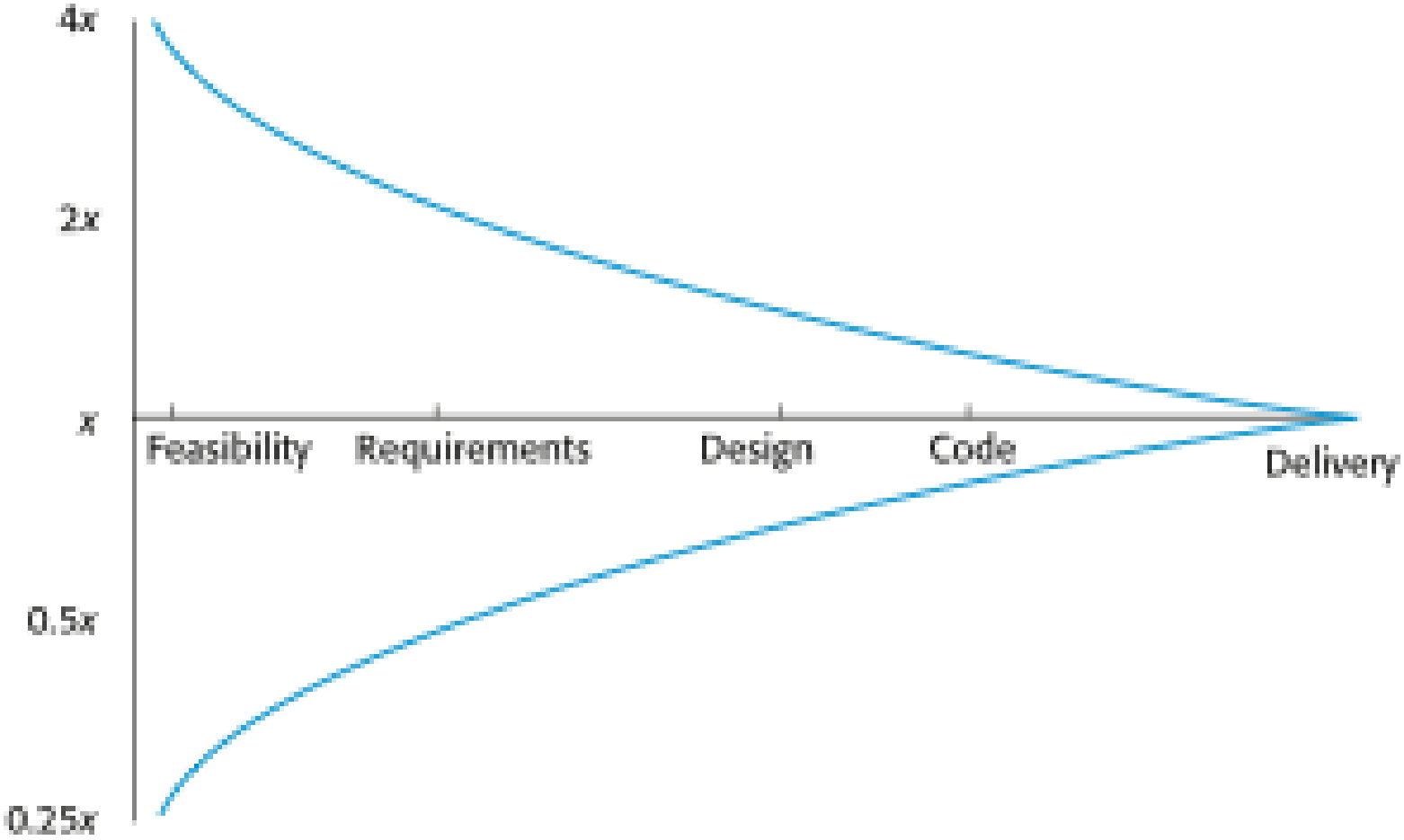
(somewhat risky way)

- 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 would complete as per the prescribed schedule; may result in a disaster

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."

Estimate uncertainty



Estimate uncertainty

Accuracies of Estimates in Other Industries vs. Software Development

Project Management Accuracy from the PMBOK Third Edition [PMI 2004]		**Software Development Accuracy from Rapid Development [McConnell 1996]**	
Conceptual	30% under to 50% over	75% under to 300% over	Initial product concept
Preliminary	20% under to 30% over	50% under to 100% over	Approved product definition
Definitive	15% under to 20% over	33% under to 50% over	Requirements specification
Control	10% under to 15% over	20% under to 25% over	Product design specification

Scheduling problems

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

– Sommerville

Estimation is among the weakest links in SE chain

sourced from *geek & poke*

Project Scheduling

Scheduling Practices

- Project management responsibilities
  - Define all project tasks
    - On large projects, hundreds of small tasks must occur to accomplish a larger goal
  - Build an activity network that depicts their interdependencies
    - Some of these tasks lie outside the mainstream and may be completed without worry of impacting on the project completion date
  - Identify the tasks that are critical within the activity network
    - if these tasks fall behind schedule, the completion date of the entire project is put into jeopardy
  - 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

Scheduling 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 Practices (continued)

- 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

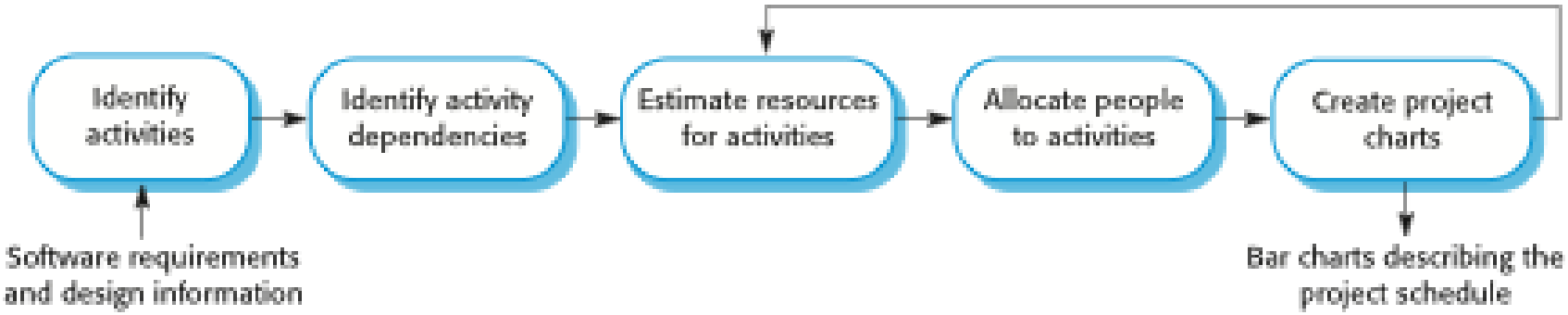
(More on next slide)

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

The Project Scheduling Process



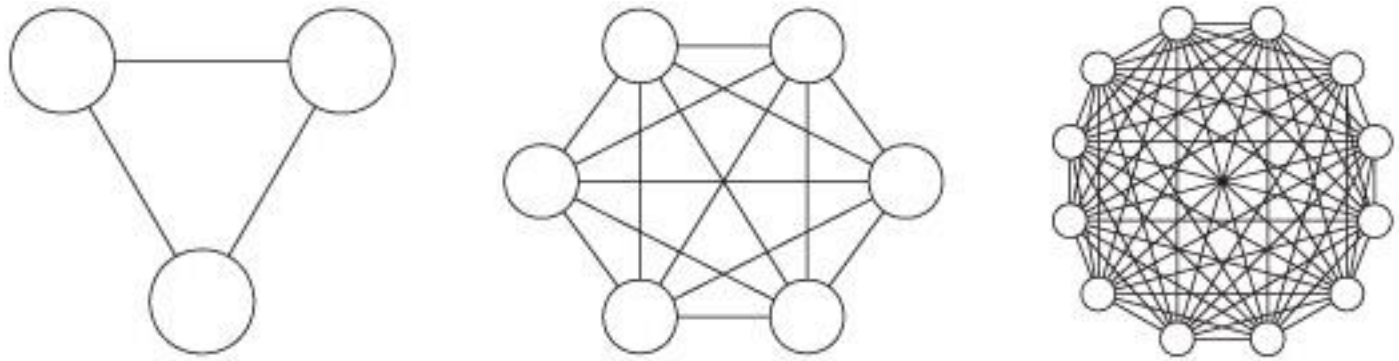
Relationship Between People and Effort

- 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

Scheduling Challenges – Team Size

Large programming projects suffer management problems that are qualitatively different than small ones because of the division of labor; that the conceptual integrity of the software product is thus critical; and that it is difficult but possible to achieve this conceptual integrity.

- *The Mythical Man-Month (Fred Brooks)*



No of communication paths explode as team size grows. Above illustrations show communication paths among teams of 3, 6, and 12

Scheduling Challenges – Productivity

There are order-of-magnitude differences among programmers and have been confirmed by many … studies of professional programmers….

- Code Complete (Steve McConnell)

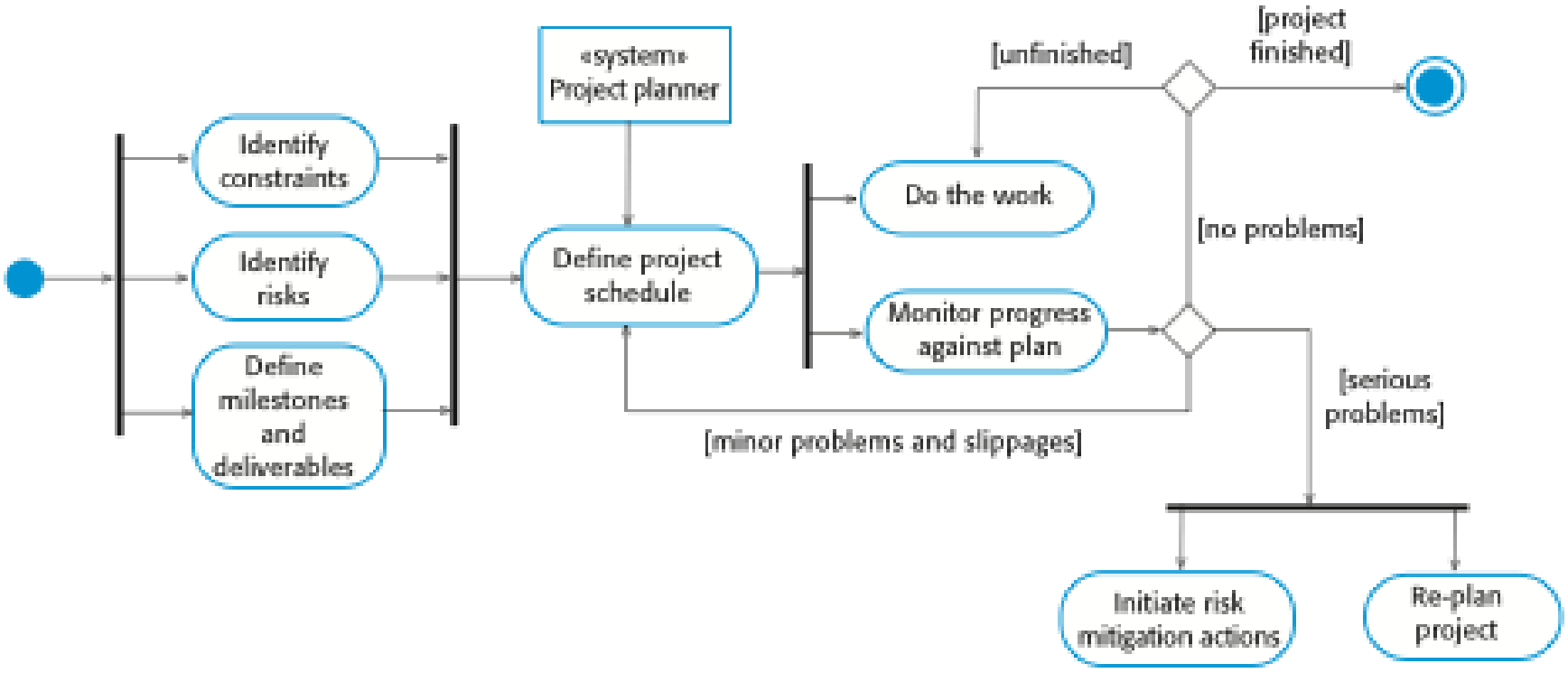
Barry Boehm [1981] makes the following recommendations for selecting team in his book *Software Engineering Economics*:

Top talent—Use better and fewer people.

Job matching—Fit the tasks to the skills and motivation of the people available.

Team balance—Select people who will complement and harmonize with each other.

The Project Planning Process



The Software Equation

A dynamic multivariable model

$$E = [LOC \times B^{0.333}/P]^3 \times (1/t^4)$$

[A 10% compression in timeline(t) leads to 52% increase in Effort/Cost(E)]

E= effort in person-months or person-years

t = project duration in months or years

B = "special skills factor"

- for small programs(5 to 15 KLOC), B=0.16
- for large programs(> KLOC), B=0.39

P = productivity parameter reflecting overall process maturity, skills, experience etc. Needs to be customized for local conditions.

- for real-time embedded software P=2000
- for system software P=10000
- for business applications P=28000

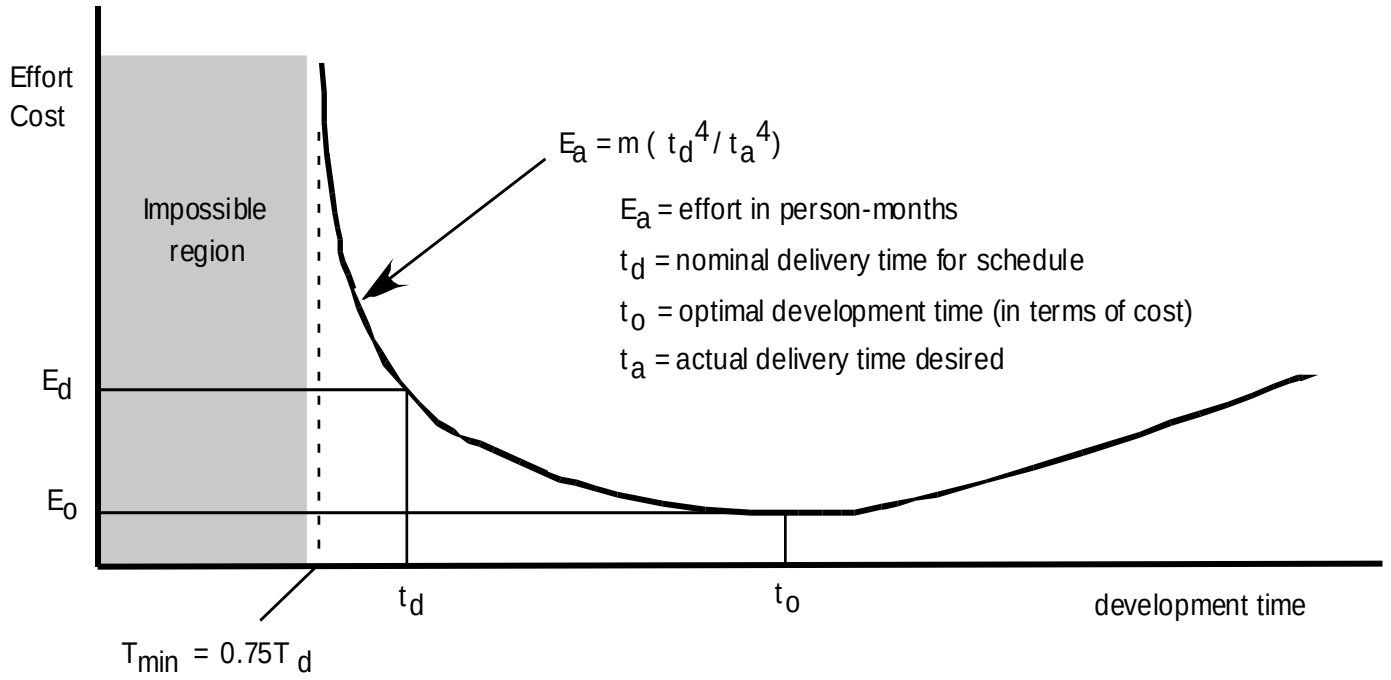
- Putnam et al.

Effort Applied vs. Delivery Time

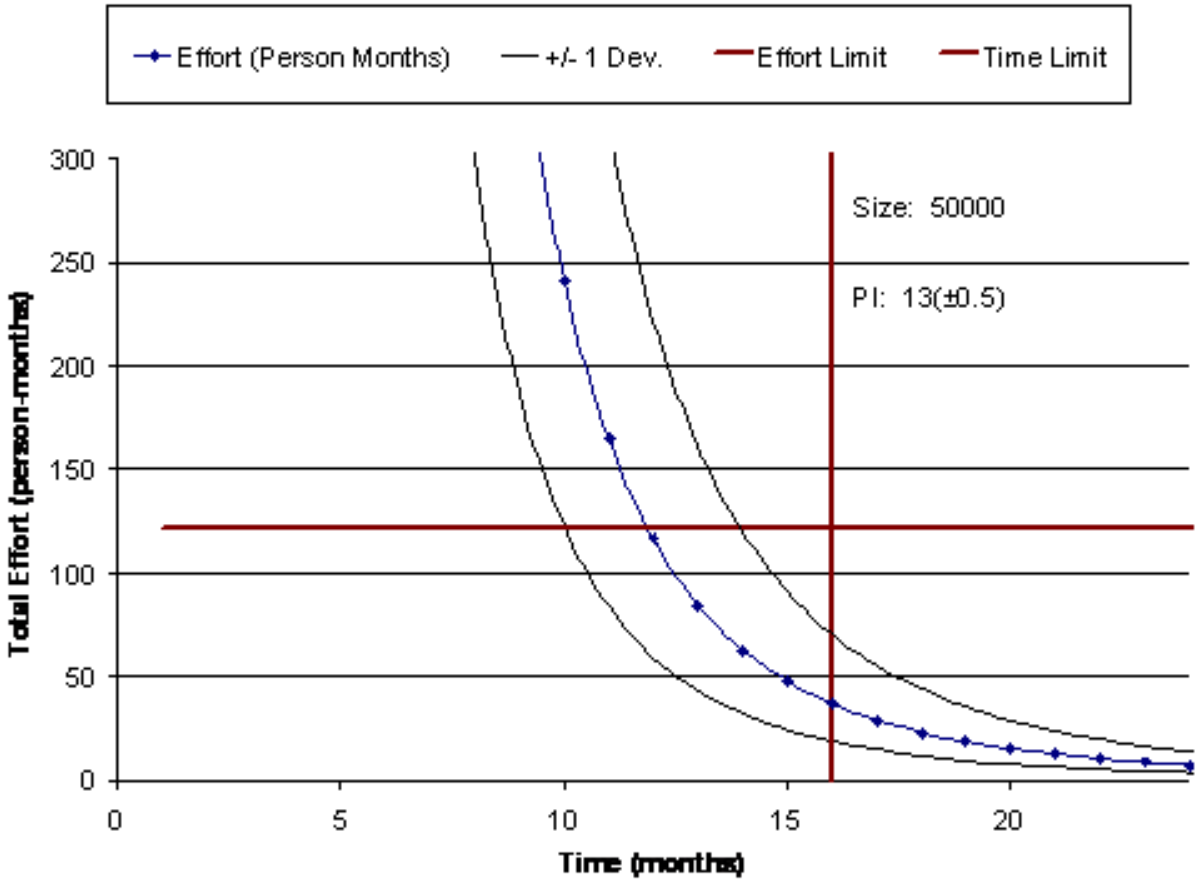
- There is a nonlinear relationship between effort applied and delivery time (Ref: Putnam-Norden-Rayleigh software equation)
  - Effort increases rapidly as the delivery time is reduced
- Also, delaying project delivery can reduce costs significantly as shown in the equation $\mathbf{E \alpha L^3/(P^3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

Effort and Delivery Time

Putnam Norden Rayleigh (PNR) curve



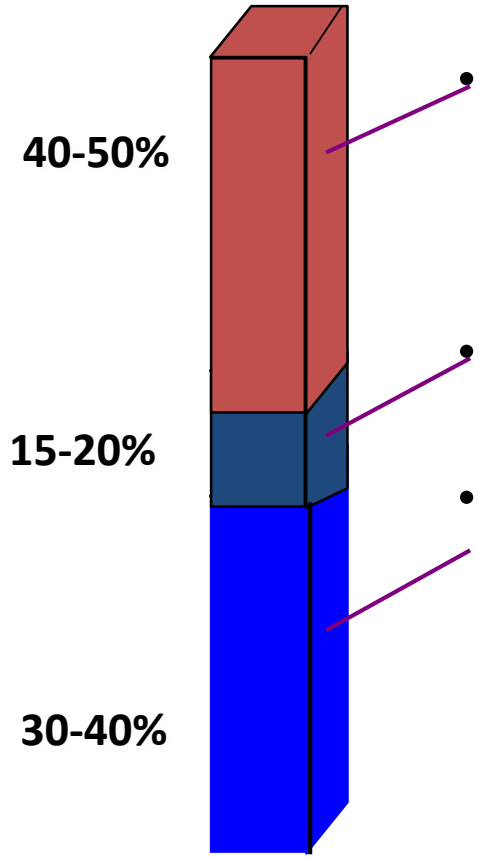
Effort vs. Time
with Varying Productivity



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

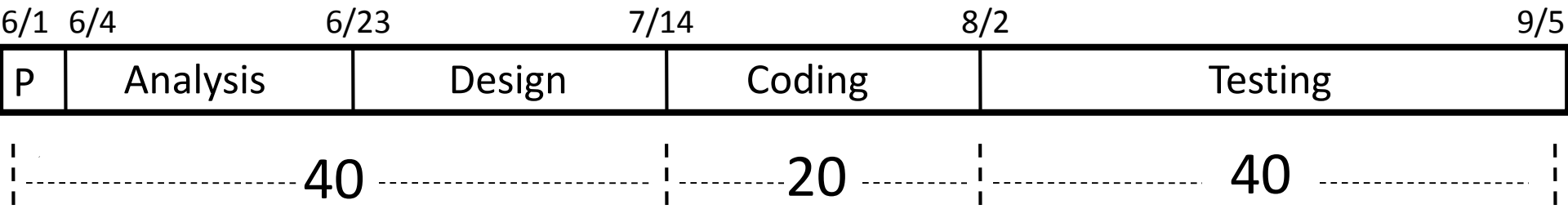
Effort Allocation



- "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

40-20-40 Distribution of Effort

Example: 100-day project



- Empirical Data
- Prescriptive process approach

A word on Film Production

40 20 40 'Guide' for project time and resources:

40% **Pre-Production** Planning

20% **Production** Filming

40% **Post-production** - editing, reflecting, promoting, screening

-Ian McCormick, Former Professor in Arts, University of Northampton

Task Network

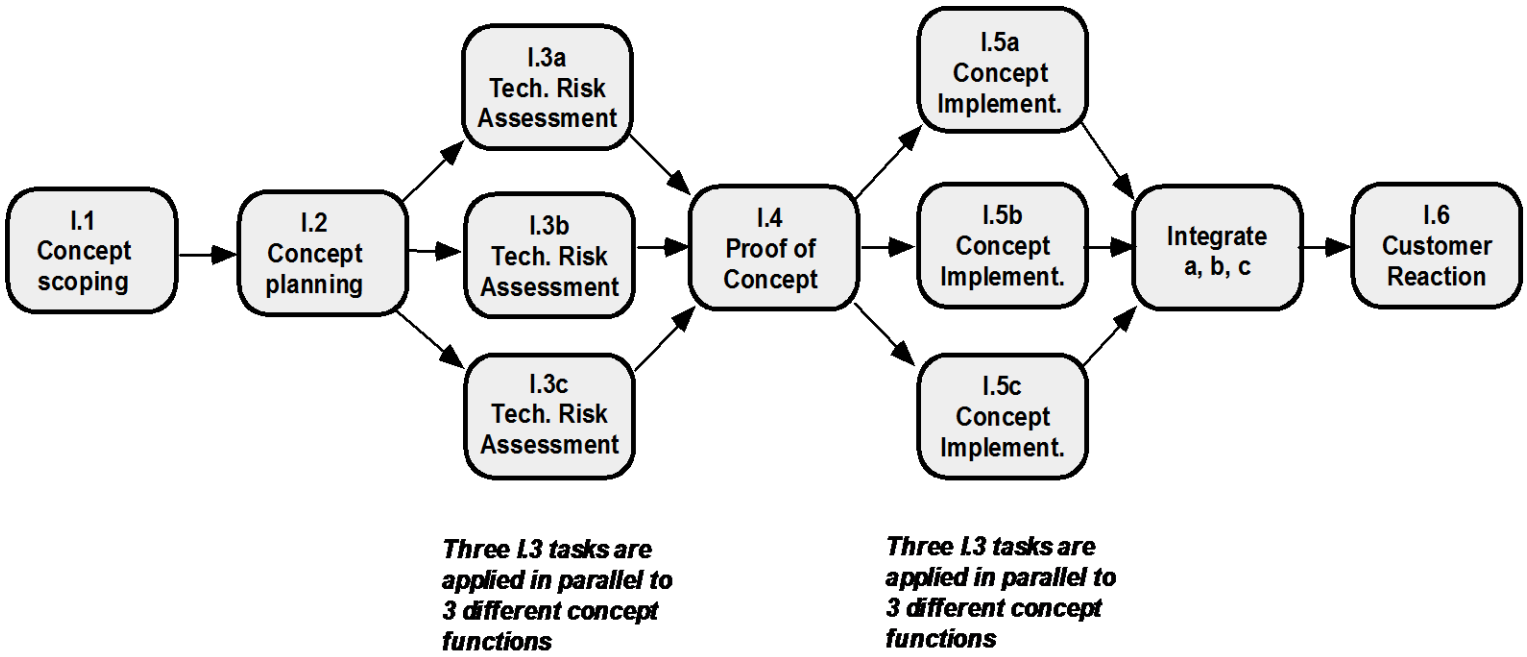
Defining a Task Set

- 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

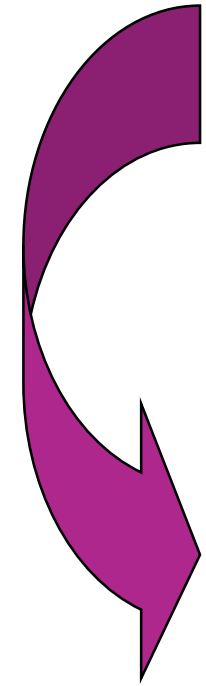
- Concept development projects
  - Explore some new business concept or application of some new technology
- New application development
  - Undertaken as a consequence of a specific customer request
- Application enhancement
  - Occur when existing software undergoes major modifications to function, 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

Define a Task Network



Concept Development Project

Task Set Refinement



is refined to

1.1 Concept scoping.

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 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
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

Technical Tasks for Object-Oriented Projects

- Technical milestone: OO analysis complete
  - All hierarchy classes defined and reviewed
  - Class attributes and operations are defined and reviewed
  - Class relationships defined and reviewed
  - Behavioral model defined and reviewed
  - Reusable classed identified
- Technical milestone: OO design complete
  - Subsystems defined and reviewed
  - Classes allocated to subsystems and reviewed
  - Task allocation has been established and reviewed
  - Responsibilities and collaborations have been identified
  - Attributes and operations have been designed and reviewed
  - Communication model has been created and reviewed

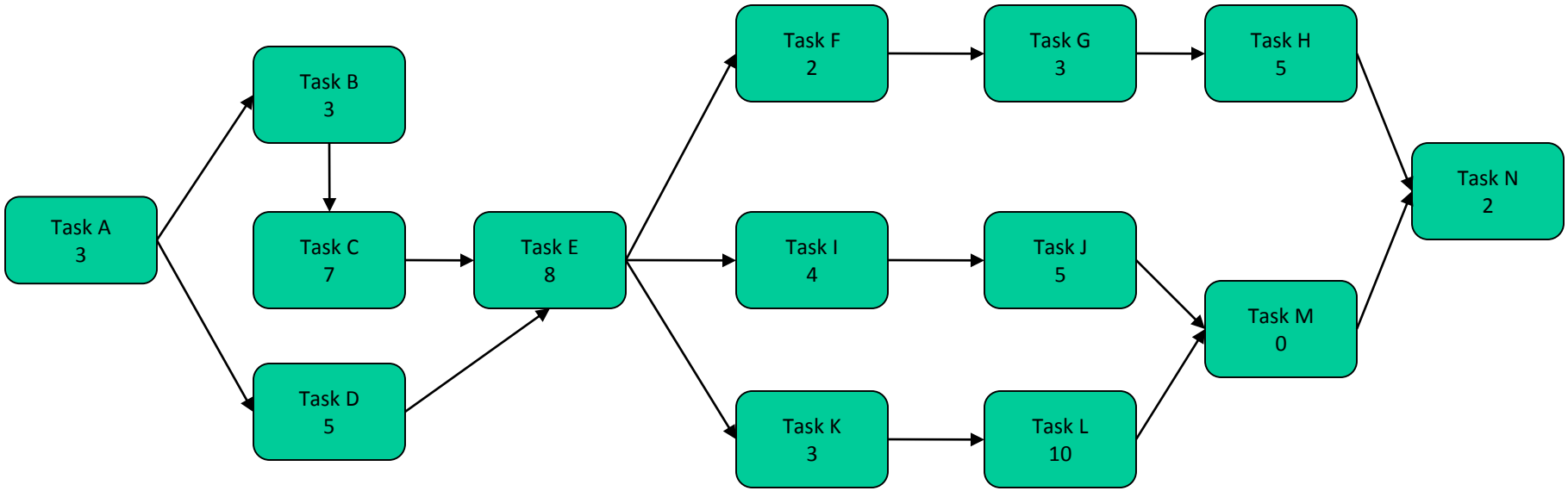
Technical Tasks for Object-Oriented Projects
(contd)

- Technical milestone: OO programming complete
  - Each new design model class has been implemented
  - Classes extracted from the reuse library have been implemented
  - Prototype or increment has been built
- Technical milestone: OO testing complete
  - The correctness and completeness of the OOA and OOD models has been reviewed
  - Class-responsibility-collaboration network has been developed and reviewed
  - Test cases are designed and class-level tests have been conducted for each class
  - Test cases are designed, cluster testing is completed, and classes have been integrated
  - System level tests are complete

Purpose of a Task Network

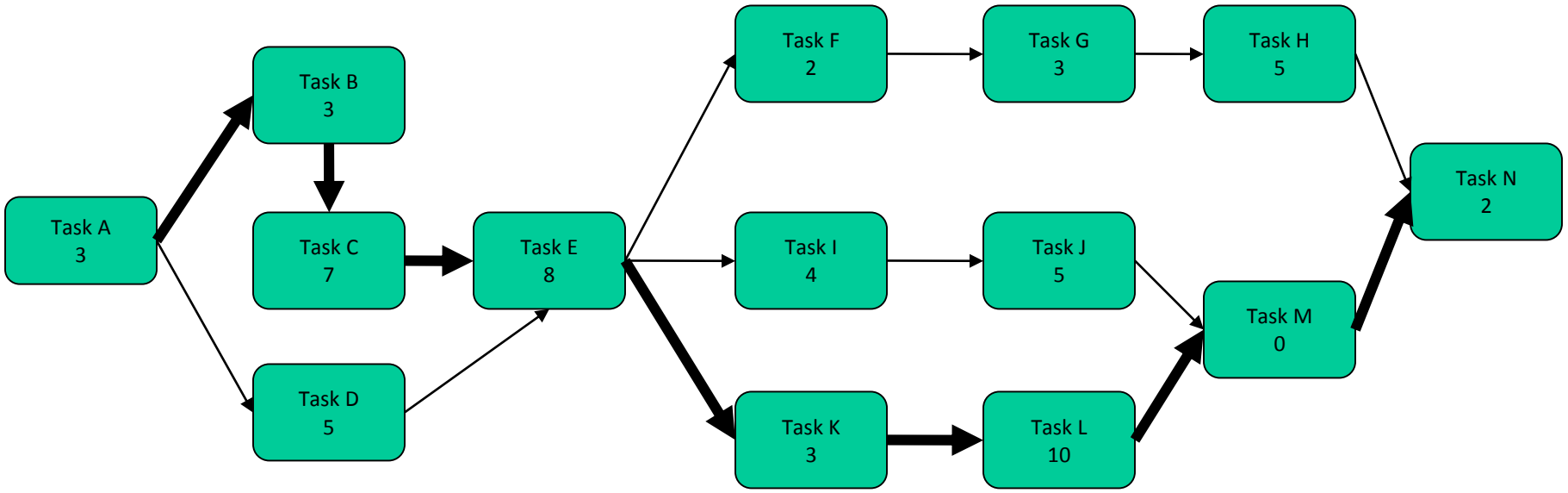
- Also called an activity network
- It is a graphic representation of the task flow for a project
- It depicts task length, sequence, concurrency, and dependency
- Points out inter-task dependencies to help the manager ensure continuous progress toward project completion
- The critical path
  - A single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

Task Network (Another Example)



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

Example Task Network with Critical Path Marked

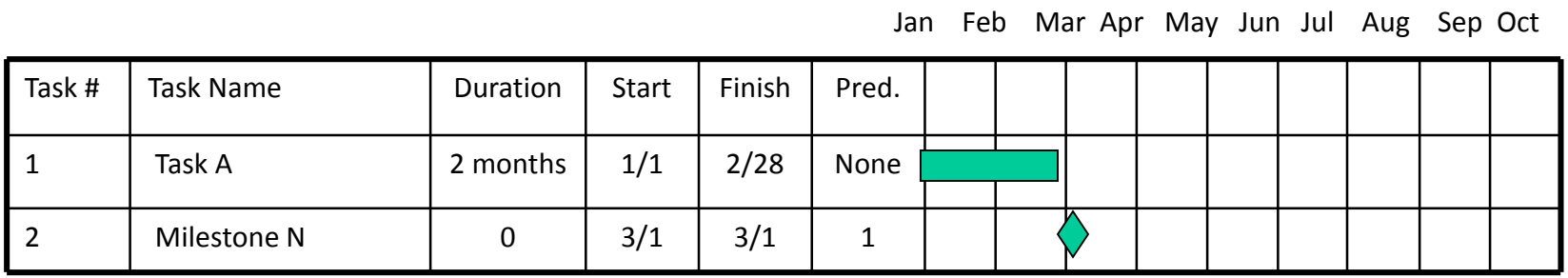


Critical path: A-B-C-E-K-L-M-N

Timeline Chart

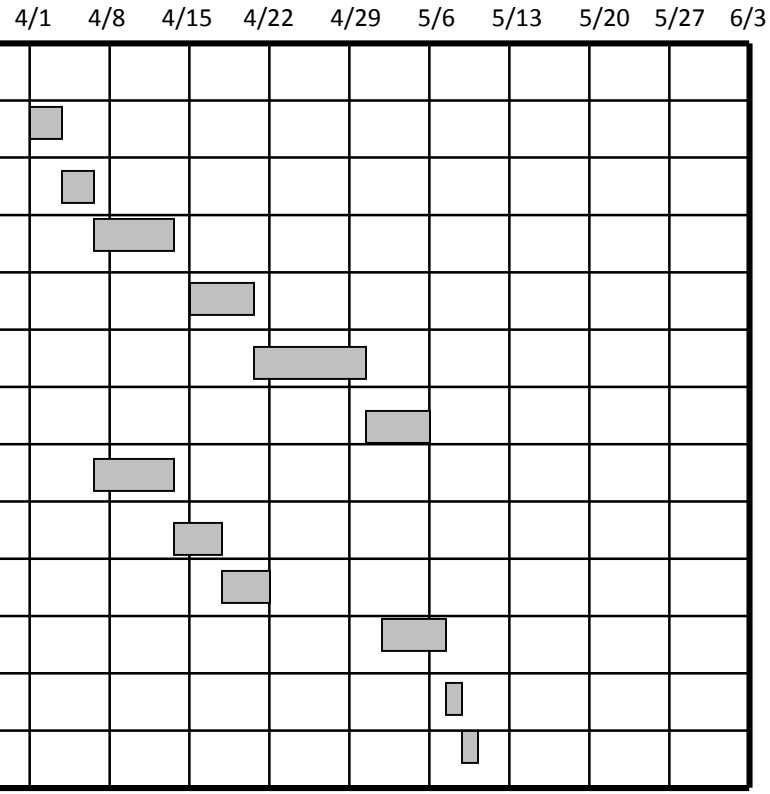
Mechanics of a Timeline Chart

- 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

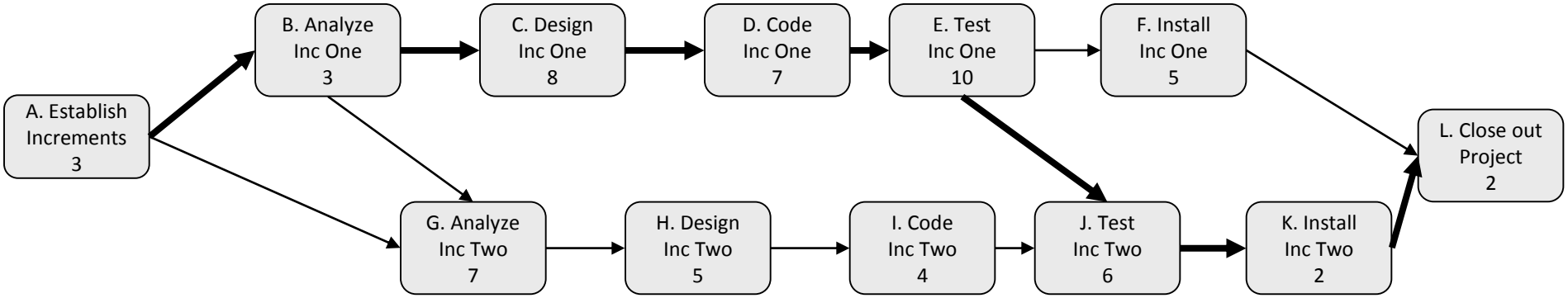


Timeline Chart and Task Networks

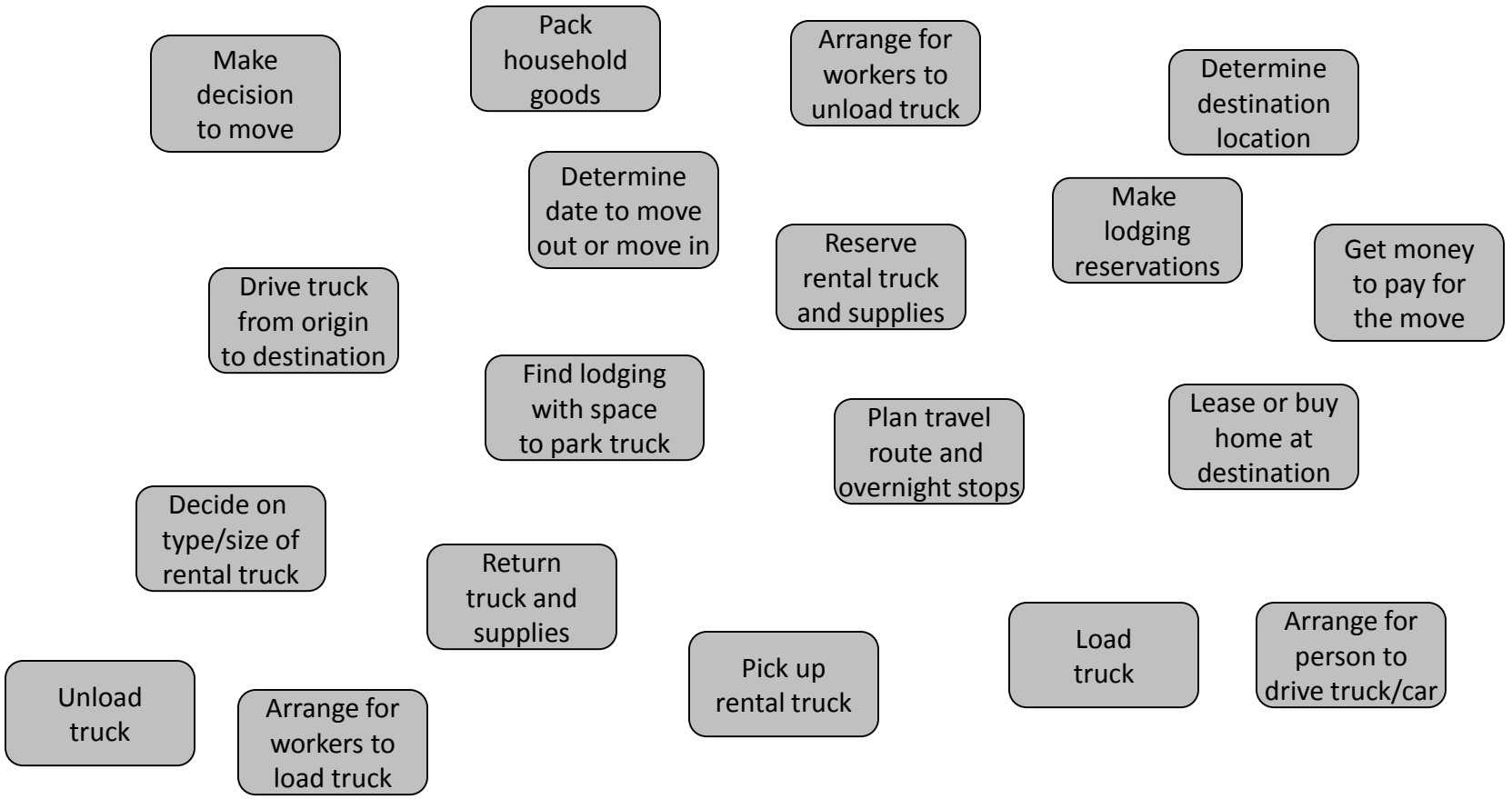
Task #	Task Name	Duration	Start	Finish	Pred.
A	Establish increments	3	4/1	4/3	None
B	Analyze Inc One	3	4/4	4/6	A
C	Design Inc One	8	4/7	4/14	B
D	Code Inc One	7	4/15	4/21	C
E	Test Inc One	10	4/22	5/1	D
F	Install Inc One	5	5/2	5/6	E
G	Analyze Inc Two	7	4/7	4/13	A, B
H	Design Inc Two	5	4/14	4/18	G
I	Code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

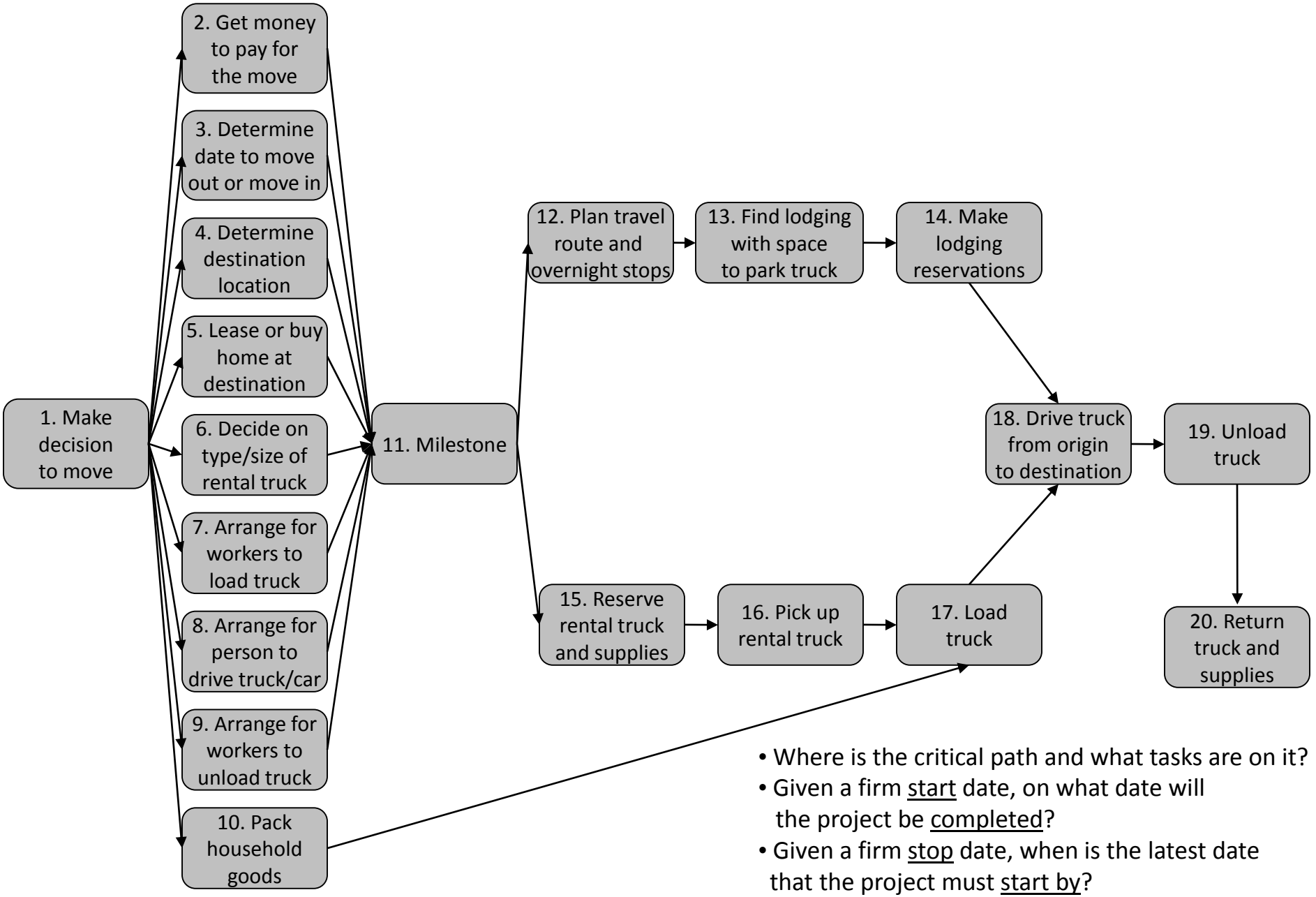


Proposed Tasks for a Long-Distance Move of 8,000 lbs of Household Goods



- 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?

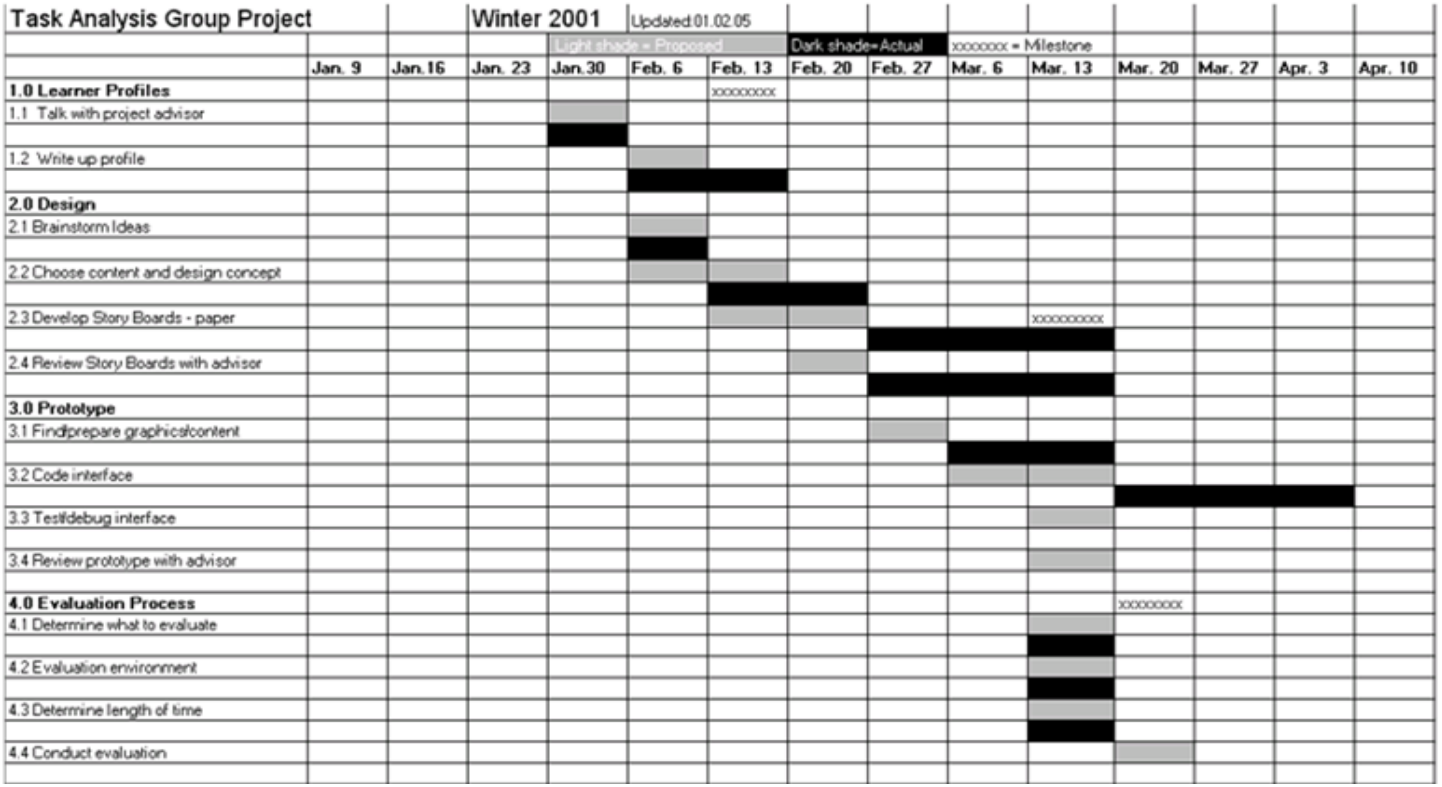
Task Network for a Long-Distance Move of 8,000 lbs of Household Goods



- 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>?

Example Timeline Chart

- Microsoft Excel can be used for Timeline chart as shown below
- Develop conventions for plan, actual, and milestones



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

Project Control

- 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

Time Boxing

- 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

Tracking 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

- Because the object-oriented process is an iterative process, each of these milestones may be revisited as different increments are delivered to the customer

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

Earned Value Analysis

How is project health measured?

- Project Manager evaluates project's triple constraint of scope, time and cost
- Key Questions
  - Is the project performing to budget?
  - Is the project on schedule to deliver the agreed scope?
- Typically Summarize Project Health using Green, Yellow, Red Status (Traffic Light Reporting)

Problems with the Traffic Light status approach

- Subjective to interpretation and influence
- No objective measurement to guide Project Health
- A project can report green and suddenly turn red before a few days before the launch
- A project can report red and be rationalized to green or yellow without any objective data
- No prior indicators to problems

Earned Value Analysis

- Earned Value Analysis (EVA)
  - Earned Value Analysis is an objective method to measure project performance in terms of scope, time and cost
  - Use EVA metrics are used to measure project health and project performance

- EVA tidbits
  - EVA used by US DOD. In 1991, Secretary of Defense canceled the Navy A-12 Avenger II Program because of performance problems detected by EVA. It was adopted by the National Aeronautics and Space Administration, United States Department of Energy and other technology-related agencies, and also many industrialized nations. (Wikipedia)

Earned Value Characteristics

- Point in Time Evaluation
- How much work did you PLAN to complete? (*Planned Value*)
- How much work did you ACTUALLY complete? (*Earned Value*)
- How much did you spend to complete the work? (*Actual Cost*)

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 (e.g., 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
  - BCWS is also referred as **PV** (Planned Value)
- Sum up the BCWS values for all work tasks to derive the budget at completion (BAC)

$$BAC = \sum (BCWS_k) \text{ for all tasks } k$$

Computing Earned Value-II

- Next, the value for *budgeted cost of work performed* (BCWP) is computed.
  - The value for BCWP is the sum of the BCWS values for all work tasks that have actually been completed by a point in time on the project schedule. Referred by many authors as **EV** (Earned Value)
- "the distinction between the BCWS and the BCWP is that the former represents the budget of the activities that were planned to be completed and the latter represents the budget of the activities that actually were completed."
- Given values for BCWS, BAC, and BCWP, important progress indicators can be computed:
  - Schedule performance index, SPI = BCWP/BCWS
  - Schedule variance, SV = BCWP – BCWS
  - SPI is an indication of the efficiency with which the project is utilizing scheduled resources.

Computing Earned Value-III

- Percent scheduled for completion = BCWS/BAC
  - provides an indication of the percentage of work that should have been completed by time *t.*
- Percent complete = BCWP/BAC
  - provides a quantitative indication of the percent of completeness of the project at a given point in time, *t.*
- *Actual cost of work performed,* ACWP, is the sum of the effort actually expended on work tasks that have been completed by a point in time on the project schedule. It is then possible to compute
    - Cost performance index, CPI = BCWP/ACWP
    - Cost variance, CV = BCWP – ACWP
    - ACWP also referred as **AC** (Actual Cost)

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

EVA Example

A $10,000 software project is scheduled for 4 weeks.

At the end of the third week, the project is 50% complete and the actual costs to date is $9,000

Planned Value (PV) = $7,500

Earned Value (EV) = $5,000

Actual Cost (AC) = $9,000

What is the project health?

Schedule Variance

= EV – PV = $5,000 – $7,500 = - $2,500

Schedule Performance Index (SPI)

= EV/PV = $5,000 / $7,500 = .66

Cost Variance

= EV – AC = $5,000 - $9,000 = - $4,000

Cost Performance Index (CPI)

= EV/AC = $5,000 / $9,000 = .55

Objective metrics indicate the project is behind schedule and over budget.

On-target projects have an SPI and CPI of 1 or greater

Forecasting Costs

- If the project continues at the current performance, what is the true cost of the project?
- Estimate At Complete

  = Budget At Complete (BAC) / CPI

  = $10,000 / .55 = $18,181

  At the end of the project, the total project costs will be $18,181

Establish Ranges to Guide Traffic Light Status

- Traffic Light status is useful in conveying overall project with one color
- Establish objective SPI and CPI ranges to determine the true project color.

Green	[1.0 - .95]
Yellow	[.94-.85]
Red	[.84, 0]

Earned Value Summary

- Earned Value is an objective method of determining project performance instead of subjective approaches
- Apply Earned Value enforces the project discipline of tracking project actual performance against baseline costs and dates
- Estimate at Complete calculation can forecast true project costs based on project performance

Summary of Project Scheduling

- To build complex software systems, many engineering tasks need to occur in parallel with one another to complete the project on time.
- The output from one task often determines when another may begin.
- Software engineers need to build activity networks that take these task interdependencies into account.
- Managers find that it is difficult to ensure that a team is working on the most appropriate tasks without building a detailed schedule and sticking to it.
- Sound project management requires that tasks are assigned to people, milestones are created, resources are allocated for each task, and progress is tracked.
- Earned value analysis is a quantitative technique for monitoring project progress