Chapter 3:
Project Management

WILEY
PowerPoint Presentation for Dennis, Wixom, & Tegarden *Systems Analysis and Design with UML, 3rd Edition*

Objectives

- Become familiar with estimation.
- Be able to create a project workplan.
- Understand why project teams use timeboxing.
- Become familiar with how to staff a project.
- Understand how computer-aided software engineering, standards, and documentation improve the efficiency of a project.
- Understand how to reduce risk on a project.

Project Management

- The discipline of planning, organizing, and managing resources to bring about the successful completion of specific project goals and objectives

Successful Projects

- Cost

  At project completion, no more money has been spent than was *originally* allocated

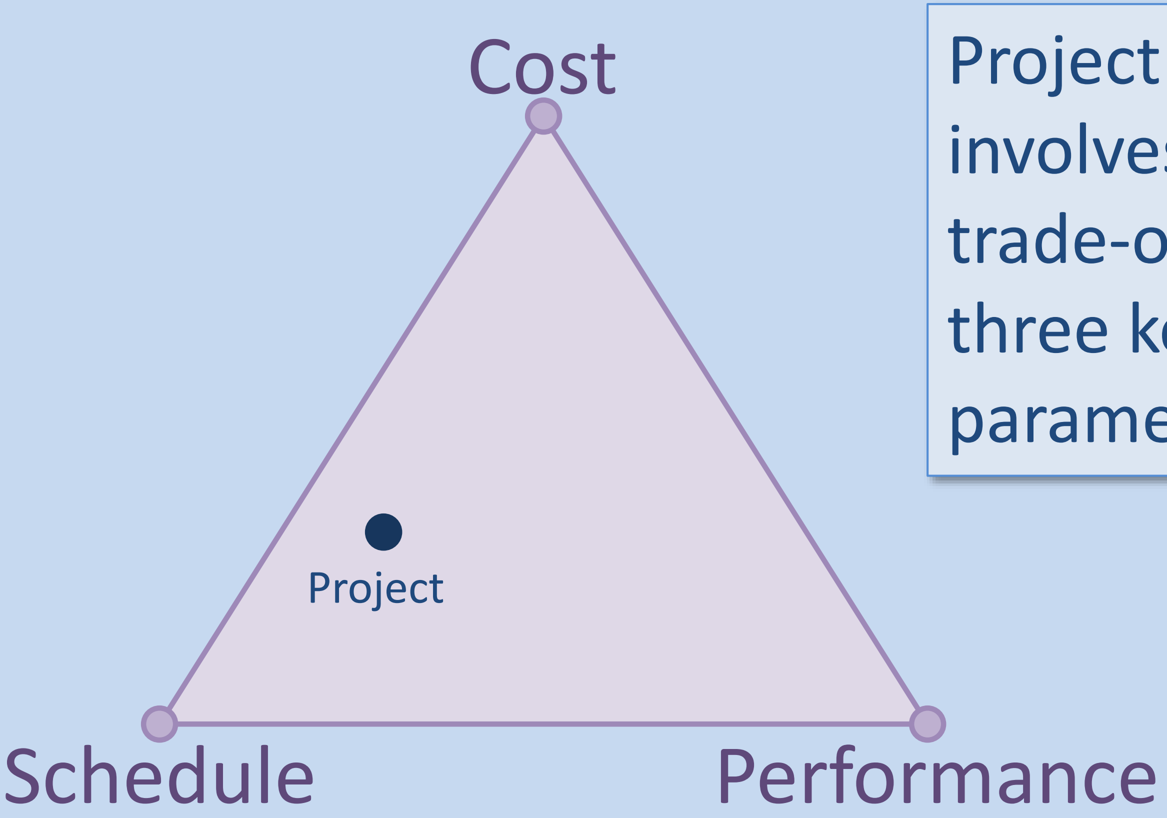
- Schedule

  The project is delivered no later than the original delivery date

- Performance

  When delivered, the project has all features and functionality that were originally required of it

Cost Schedule Performance Trade-offs



Project management involves balancing trade-offs among the three key project parameters

Project Management

- A critical factor for success is to
  - Start with realistic assessment of the work
  - And then manage the project according to that assessment

Project Management

- Steps
  - Identifying project size
  - Creating and managing the workplan
  - Staffing the project
  - Coordinating project activities

Identifying Project Size

Estimating Project Size

- Estimation sources
  - Can be provided with the methodology that is used
  - Taken from projects with similar tasks and technologies
  - Provided by experienced developers

Estimating Project Size

- A good practice is to keep track of actual values for time and effort, so that real data can be used for future projects

Estimating Project Size

- Estimation methods
  - Function point analysis (FPA)
  - Use-case points
  - A task-decomposition using work breakdown structures (WBS)
  - Time-boxing
  - …

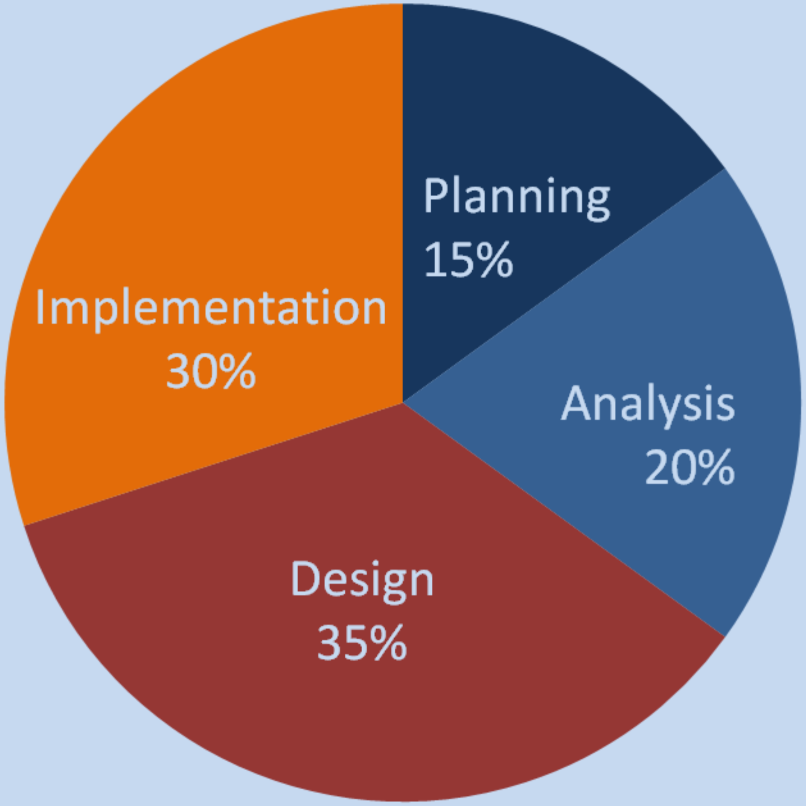
Estimating Project Size

- Simplest method uses the amount of time spent in the planning phase to predict the time required for entire project

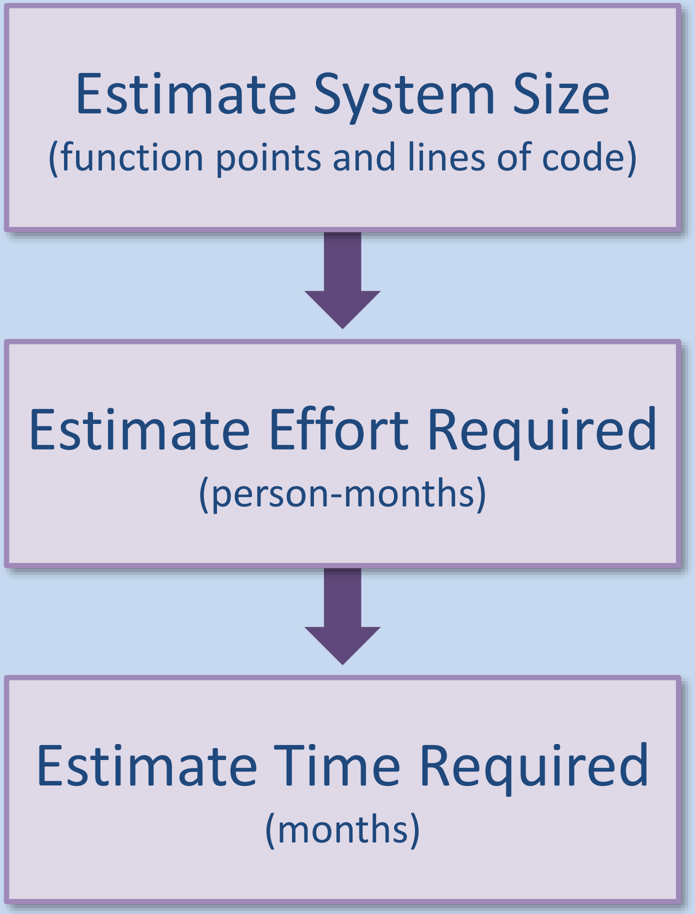
Estimating Project Timeframes – Industry Standards

Planning

System Development Life Cycle



Function Point Approach



Estimate System Size

(function points and lines of code)

Estimate Effort Required

(person-months)

Estimate Time Required

(months)

Function Point Approach

- Function point
  - – Developed in 1979 by Allen Albrecht of IBM
  - – A measure of program size based on system's number and complexity of inputs, outputs, queries, files and program interfaces

Function Point Approach (Estimating System Size)

- Steps
  1. Calculate Total Unadjusted Function Points (TUFP)
  2. Calculate Adjusted Processing Complexity (APC) or use an APC ranges from 0.65 for very simple systems, 1.0 for normal systems and 1.35 for complex systems
  3. Calculate Total Adjusted Function Points (TAFP) by multiplying APC by TUFP
  4. Calculate Lines of Code (LOC) by multiplying TAFP by table value for the selected programming language

Function Point Approach (TUFP)

Description	Total Number	Complexity			Total
		Low	Medium	High	
Inputs	6	3 x 3	2 x 4	1 x 6	23
Outputs	19	4 x 4	10 x 5	5 x 7	101
Queries	10	7 x 3	0 x 4	3 x 6	39
Files	15	0 x 7	15 x 10	0 x 15	150
Program Interfaces	3	1 x 5	0 x 7	2 x 10	25
Total Unadjusted Function Points (TUFP)					**338**

Function Point Approach (APC)

Data Communications	3
Heavy use configuration	0
Transaction rate	0
End-user efficiency	0
Complex processing	0
Installation ease	0
Multiple sites	0
Performance	0
Distributed functions	2
Online data entry	2
Online update	0
Reusability	0
Operational ease	0
Extensibility	0
Total Processing Complexity	**7**

0 = no effect on processing complexity; 5 = great effect on processing complexity

Function Point Approach (APC)

- APC = 0.65 + (0.01 x 7) = 0.72
- Or use a APC from 0.65 to 1.35 as stated before

Function Point Approach (TAPC)

- TAPC = 0.72 (APC) x 338 (TUFP) = 243

Function Point Approach (LOC)

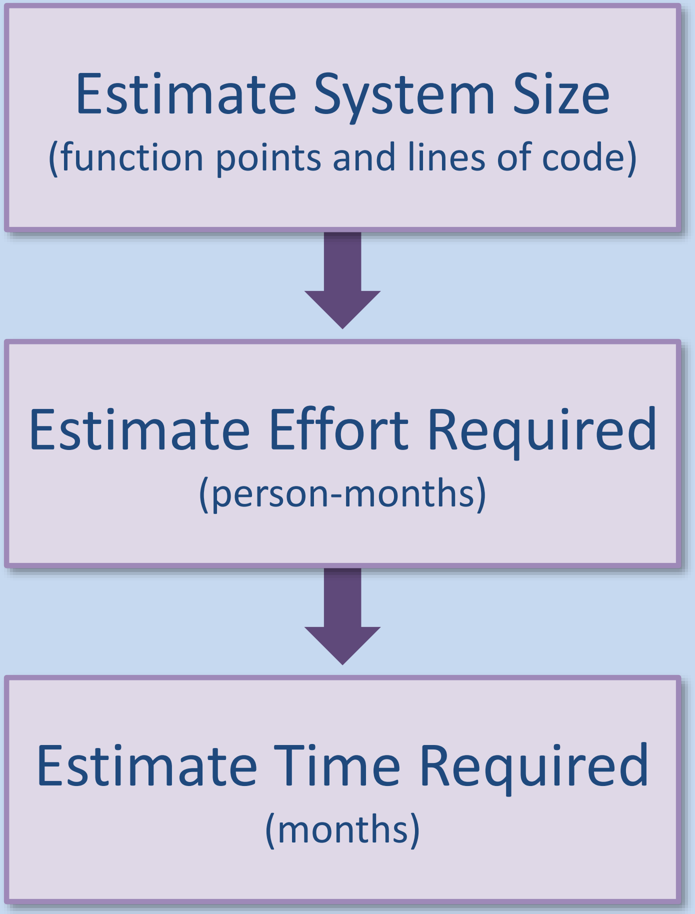
Language	Average Number of Lines of Code per Function Point
Assembler	119
ASP	51
C	97
C++	50
C#	54
COBOL	61
HTML	34
J2EE	46
Java	53
Javascript	47
Lotus Notes	23
Perl	24
VB.NET	52
SQL	21
Visual Basic	42

Source: QSM (http://www.qsm.com/resources/function-point-languages-table)

Function Point Approach (LOC)

- If VB is used for this example project LOC=42*243 = 10,206 lines of code
- If Java is used for this example project LOC=53*243 = 12,879 lines of code

Function Point Approach



Estimate System Size

(function points and lines of code)

Estimate Effort Required

(person-months)

Estimate Time Required

(months)

Function Point Approach (Estimating Effort Required)

- Effort is a function of system size combined with production rates
- COCOMO model developed by Boehm
- There are different versions of COCOMO that vary based on complexity, size, experience of developers and the type of SW

Function Point Approach (Estimating Effort Required)

- E.g. For small-to-moderate sized business software projects (i.e., 100,000 lines of code and ten or fewer programmers), model is
  - effort (in person-months) = 1.4 * thousands of lines of code
- For example, to develop a system requiring 10,000 lines of code, effort is 14 person-month

Function Point Approach (Estimating Time Required)

- Can use historical data or estimation software
- One rule of thumb is
  - schedule time (months) = 3.0 * person-months$^{1/3}$
- This estimate is for the analysis, design and implementation, does not include planning

Creating and Managing the Workplan

PowerPoint Presentation for Dennis, Wixom, & Tegarden *Systems Analysis and Design with UML, 3rd Edition*

Developing Work Plans

- A *work plan*, is a dynamic schedule that records and keeps track of all **tasks** to be accomplished over the course of the project

Developing Work Plans

- Created after a project manager has a general idea of the project's size and rough schedule
- The work plan is usually the main item in a project management software application

Sample Task

Task name:	Perform economic feasibility
Start date:	Jan 5, 2010
Completion date:	Jan 19, 2010
Person assigned to the task:	Mary Smith (project sponsor)
Deliverable(s):	Cost-benefit analysis
Completion status:	Complete
Priority:	High
Resources needed:	Spreadsheet software
Estimated time:	16 hours
Actual time:	14.5 hours

Identifying Tasks

- Top-down approach
  - Identify highest level tasks
  - Break them into increasingly smaller units

Identifying Tasks

- Methodology
  - Using standard list of tasks and modifying it as necessary
    - Most organizations have a methodology they use for projects

Work Breakdown Structure (WBS)

- Defines all of the tasks to be accomplished during the project in terms of a hierarchical structure

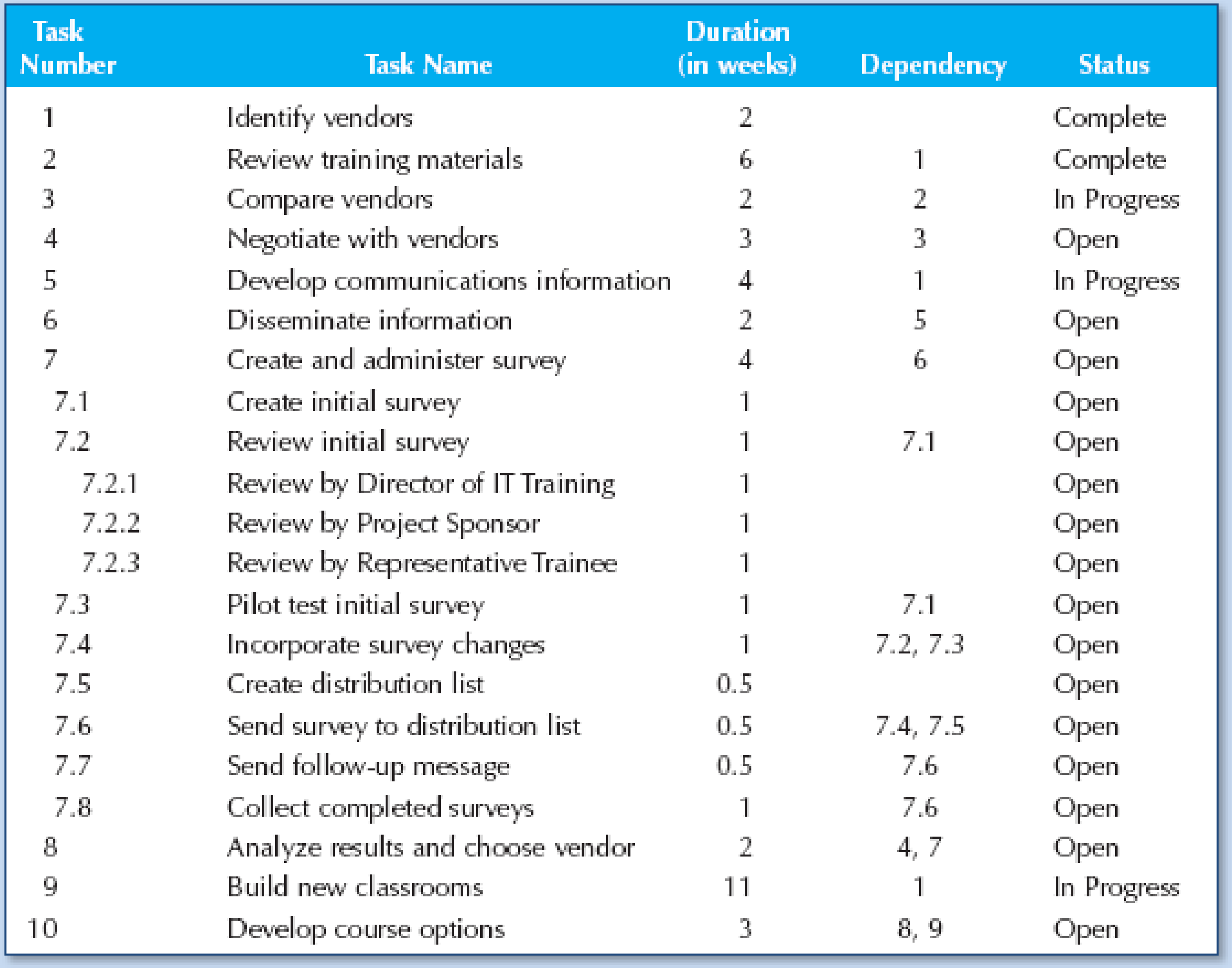
From: Systems Engineering: Principles and Practice, Kossiakoff and Sweet

Work Breakdown Structure (WBS)

- Typically defines the whole system to be developed, produced, tested, deployed, and supported including hardware, software, services and data
- Defines a skeleton or framework on which the project is to be implemented

From: Systems Engineering: Principles and Practice, Kossiakoff and Sweet

Work Breakdown Structure



Task Number	Task Name	Duration (in weeks)	Dependency	Status
1	Identify vendors	2		Complete
2	Review training materials	6	1	Complete
3	Compare vendors	2	2	In Progress
4	Negotiate with vendors	3	3	Open
5	Develop communications information	4	1	In Progress
6	Disseminate information	2	5	Open
7	Create and administer survey	4	6	Open
7.1	Create initial survey	1		Open
7.2	Review initial survey	1	7.1	Open
7.2.1	Review by Director of IT Training	1		Open
7.2.2	Review by Project Sponsor	1		Open
7.2.3	Review by Representative Trainee	1		Open
7.3	Pilot test initial survey	1	7.1	Open
7.4	Incorporate survey changes	1	7.2, 7.3	Open
7.5	Create distribution list	0.5		Open
7.6	Send survey to distribution list	0.5	7.4, 7.5	Open
7.7	Send follow-up message	0.5	7.6	Open
7.8	Collect completed surveys	1	7.6	Open
8	Analyze results and choose vendor	2	4, 7	Open
9	Build new classrooms	11	1	In Progress
10	Develop course options	3	8, 9	Open

Example list of high-level tasks to implement a new IT training class

Gantt Chart

- A type of bar chart that illustrates a project schedule
- Illustrate the start and finish dates of the terminal elements and summary elements of a project

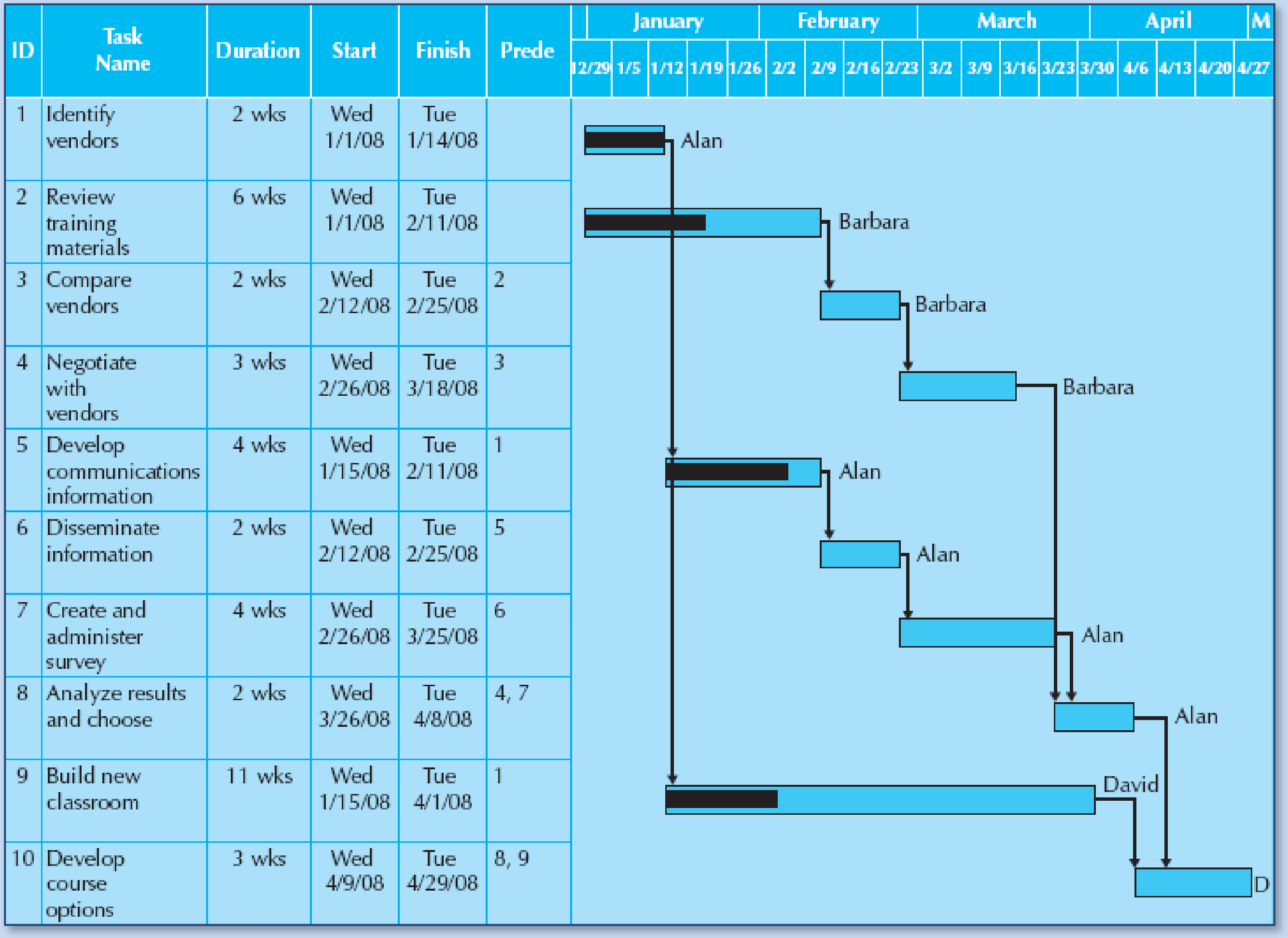
From: wikipedia.org

Gantt Chart

- Terminal elements and summary elements comprise the WBS of the project.
- Some Gantt charts also show the dependency (i.e., precedence network) relationships between activities

From: wikipedia.org

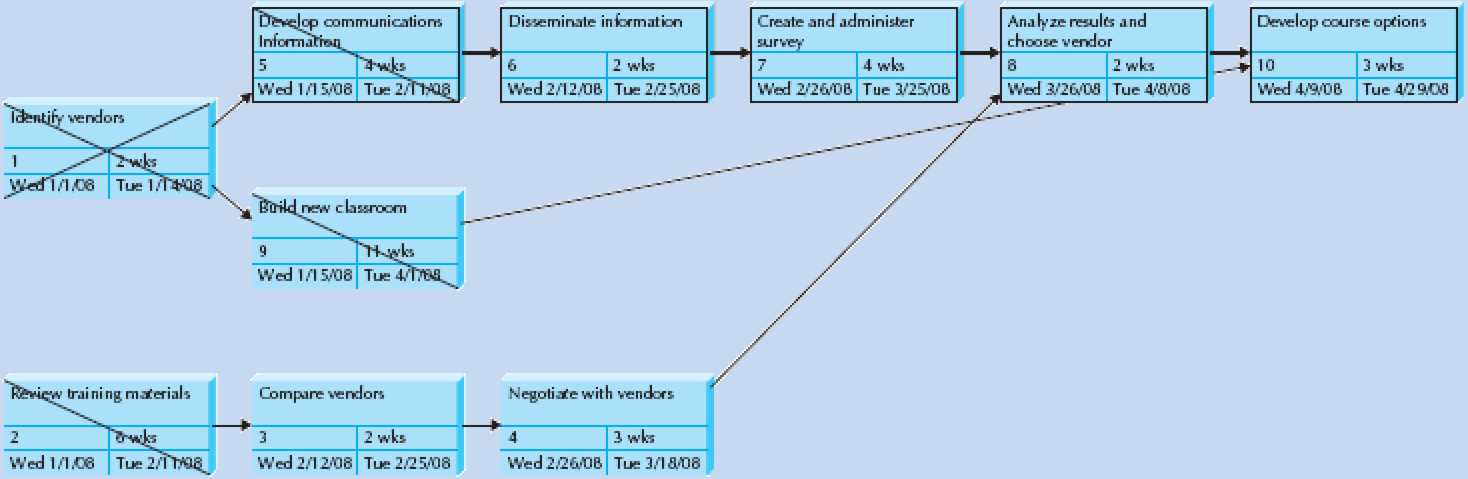
Gantt Chart



ID	Task Name	Duration	Start	Finish	Prede
1	Identify vendors	2 wks	Wed 1/1/08	Tue 1/14/08	
2	Review training materials	6 wks	Wed 1/1/08	Tue 2/11/08	
3	Compare vendors	2 wks	Wed 2/12/08	Tue 2/25/08	2
4	Negotiate with vendors	3 wks	Wed 2/26/08	Tue 3/18/08	3
5	Develop communications information	4 wks	Wed 1/15/08	Tue 2/11/08	1
6	Disseminate information	2 wks	Wed 2/12/08	Tue 2/25/08	5
7	Create and administer survey	4 wks	Wed 2/26/08	Tue 3/25/08	6
8	Analyze results and choose	2 wks	Wed 3/26/08	Tue 4/8/08	4, 7
9	Build new classroom	11 wks	Wed 1/15/08	Tue 4/1/08	1
10	Develop course options	3 wks	Wed 4/9/08	Tue 4/29/08	8, 9

Pert (Program Evaluation and Review Technique) Chart

- Used to communicate task dependencies
- Allows easier visualization of tasks on a critical path



PowerPoint Presentation for Dennis, Wixom, & Tegarden *Systems Analysis and Design with UML, 3rd Edition*

WBS, CPM and Gantt Charts

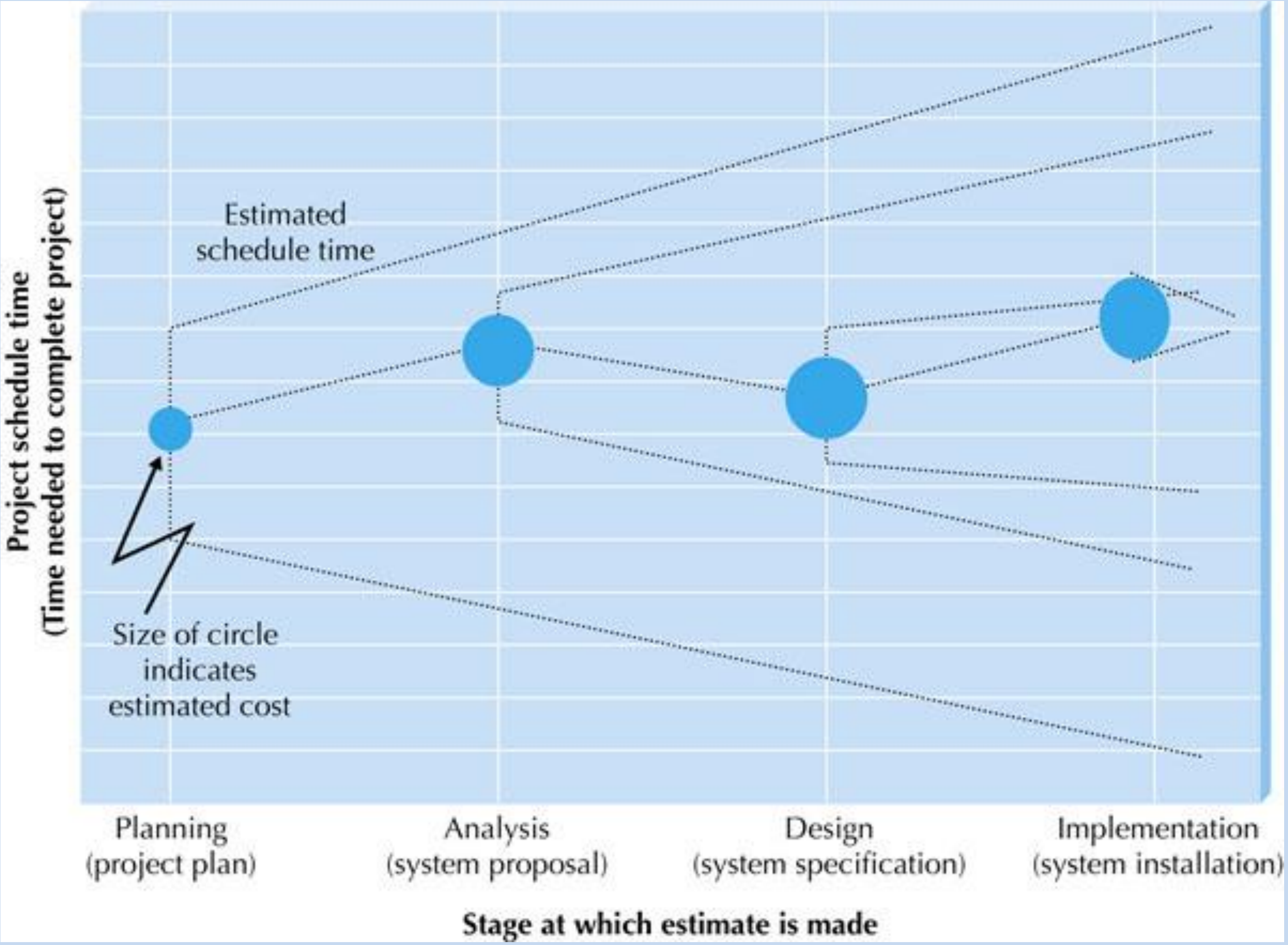
- WBS must be defined first
- Avoid defining WBS and schedule at the same time which is a common error

From: Systems Engineering: Principles and Practice, Kossiakoff and Sweet

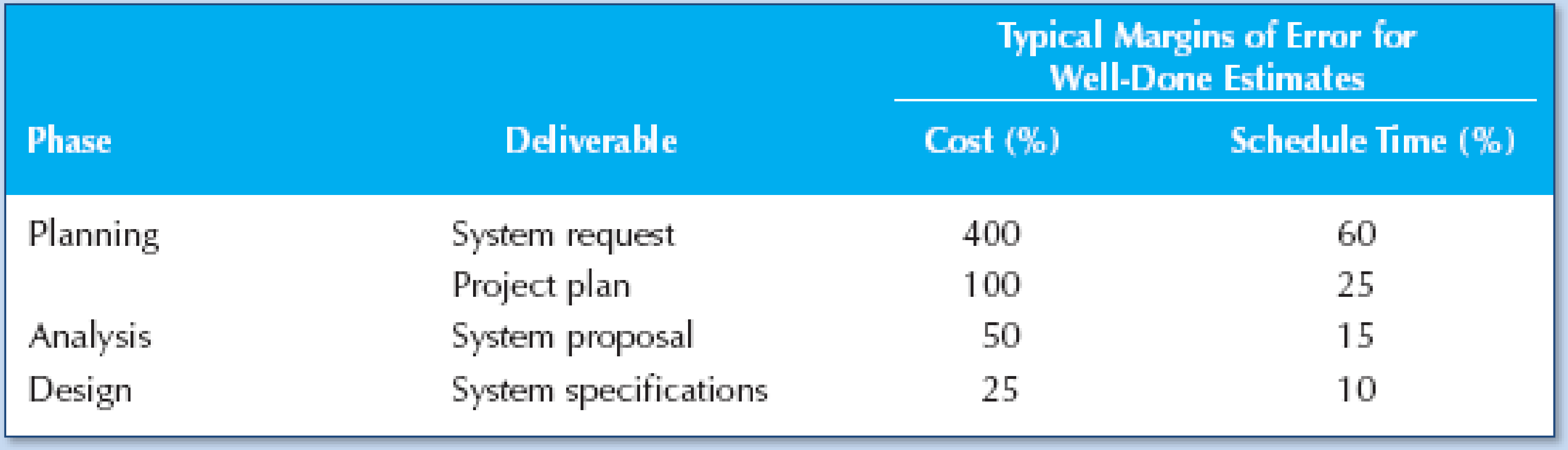
Refining Estimates

- The estimates produced during planning need to be refined as the project progresses
- It is virtually impossible to develop an exact assessment of project’s schedule before analysis and design are conducted

Refining Estimates

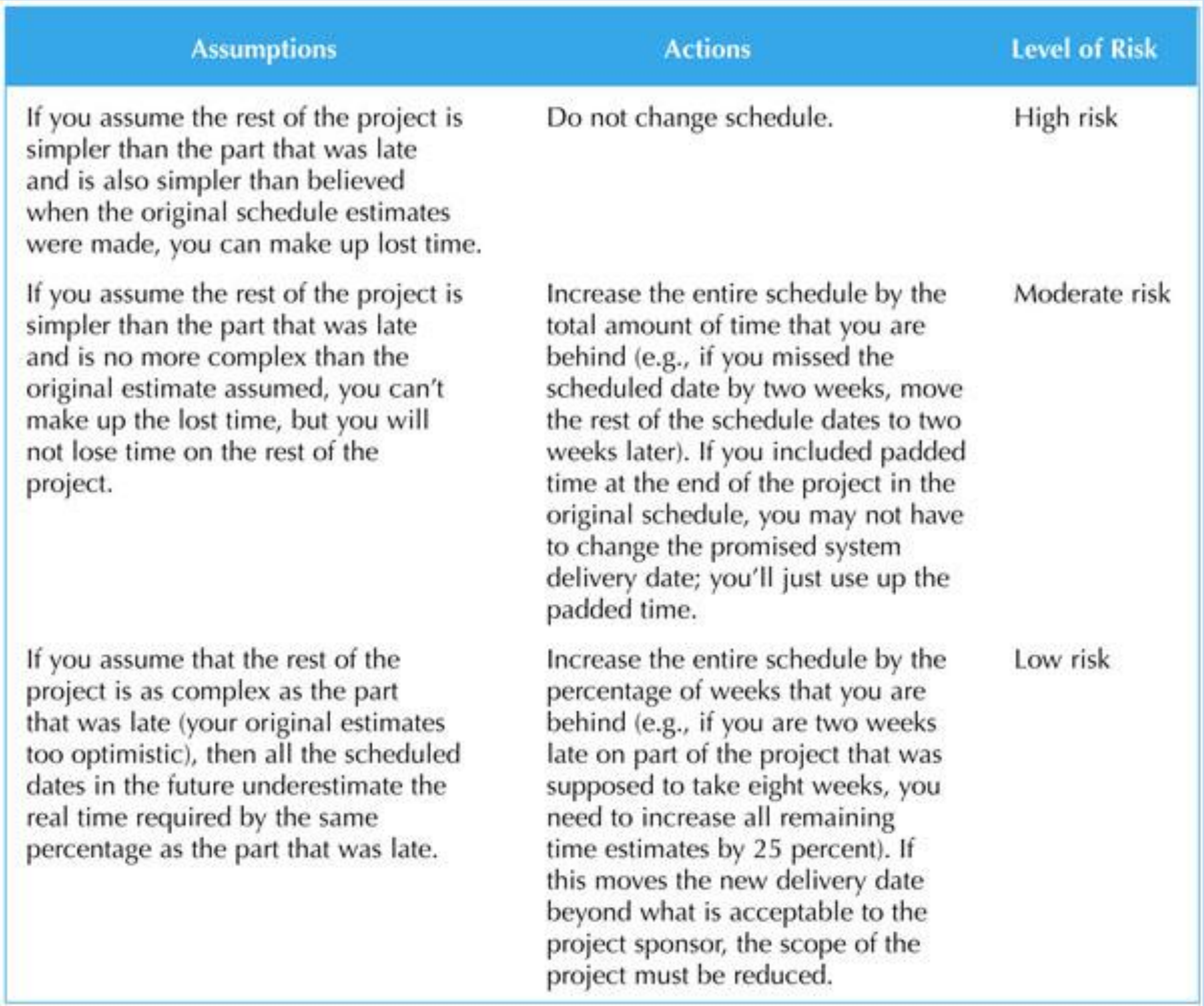

Estimated schedule time
Size of circle indicates estimated cost
Project schedule time (Time needed to complete project)
Planning (project plan)
Analysis (system proposal)
Design (system specification)
Implementation (system installation)
Stage at which estimate is made

Refining Estimates



		Typical Margins of Error for Well-Done Estimates	
Phase	Deliverable	Cost (%)	Schedule Time (%)
Planning	System request	400	60
	Project plan	100	25
Analysis	System proposal	50	15
Design	System specifications	25	10

Refining Estimates (Possible Actions when a Schedule Date is Missed)



Assumptions	Actions	Level of Risk
If you assume the rest of the project is simpler than the part that was late and is also simpler than believed when the original schedule estimates were made, you can make up lost time.	Do not change schedule.	High risk
If you assume the rest of the project is simpler than the part that was late and is no more complex than the original estimate assumed, you can't make up the lost time, but you will not lose time on the rest of the project.	Increase the entire schedule by the total amount of time that you are behind (e.g., if you missed the scheduled date by two weeks, move the rest of the schedule dates to two weeks later). If you included padded time at the end of the project in the original schedule, you may not have to change the promised system delivery date; you'll just use up the padded time.	Moderate risk
If you assume that the rest of the project is as complex as the part that was late (your original estimates too optimistic), then all the scheduled dates in the future underestimate the real time required by the same percentage as the part that was late.	Increase the entire schedule by the percentage of weeks that you are behind (e.g., if you are two weeks late on part of the project that was supposed to take eight weeks, you need to increase all remaining time estimates by 25 percent). If this moves the new delivery date beyond what is acceptable to the project sponsor, the scope of the project must be reduced.	Low risk

Scope Management

- Scope creep happens when new requirements are added to the project after the original project scope was defined and “frozen”

Scope Management

- Users may demand new functionality or managers may decide that the system should support a new strategy
- After the project begins, it becomes increasingly difficult to address changing requirements

Scope Management

- Keys are
  - to identify requirements as well as possible in the beginning
  - To apply analysis techniques effectively

Scope Management

- Some requirements may be missed no matter what precautions are taken
  - Only absolutely necessary requirements may be added after project begins (by altering project deadline)
  - Or they may be recorded as future enhancements to be added in future releases

Timeboxing

- Another approach to scope management
- Instead of task oriented approach, some companies use time oriented approach

Timeboxing

- Meeting a deadline is more important than delivering functionality
- Delivering the core of the system in specified deadline is better than extending deadline for functionalities that will not be used much

Timeboxing Steps

1. Set the date for system delivery
2. Prioritize the functionality that needs to be included in the system
3. Build the core of the system (the functionality ranked as most important)
4. Postpone functionality that cannot be provided within the time frame
5. Deliver the system with core functionality
6. Repeat steps 3 through 5 to add refinements and enhancements

Evolutionary Work Breakdown Structures and Iterative Workplans

- OO systems approaches uses incremental and iterative development
  - Project planning for OO systems also requires an incremental and iterative process
- Evolutionary WBSs allow the analyst to develop an iterative workplan

Evolutionary Work Breakdown Structures and Iterative Workplans

- Are organized in a standard manner across all projects: by workflows, phases and tasks
  - Prevents prematurely committing to a specific architecture of a new system

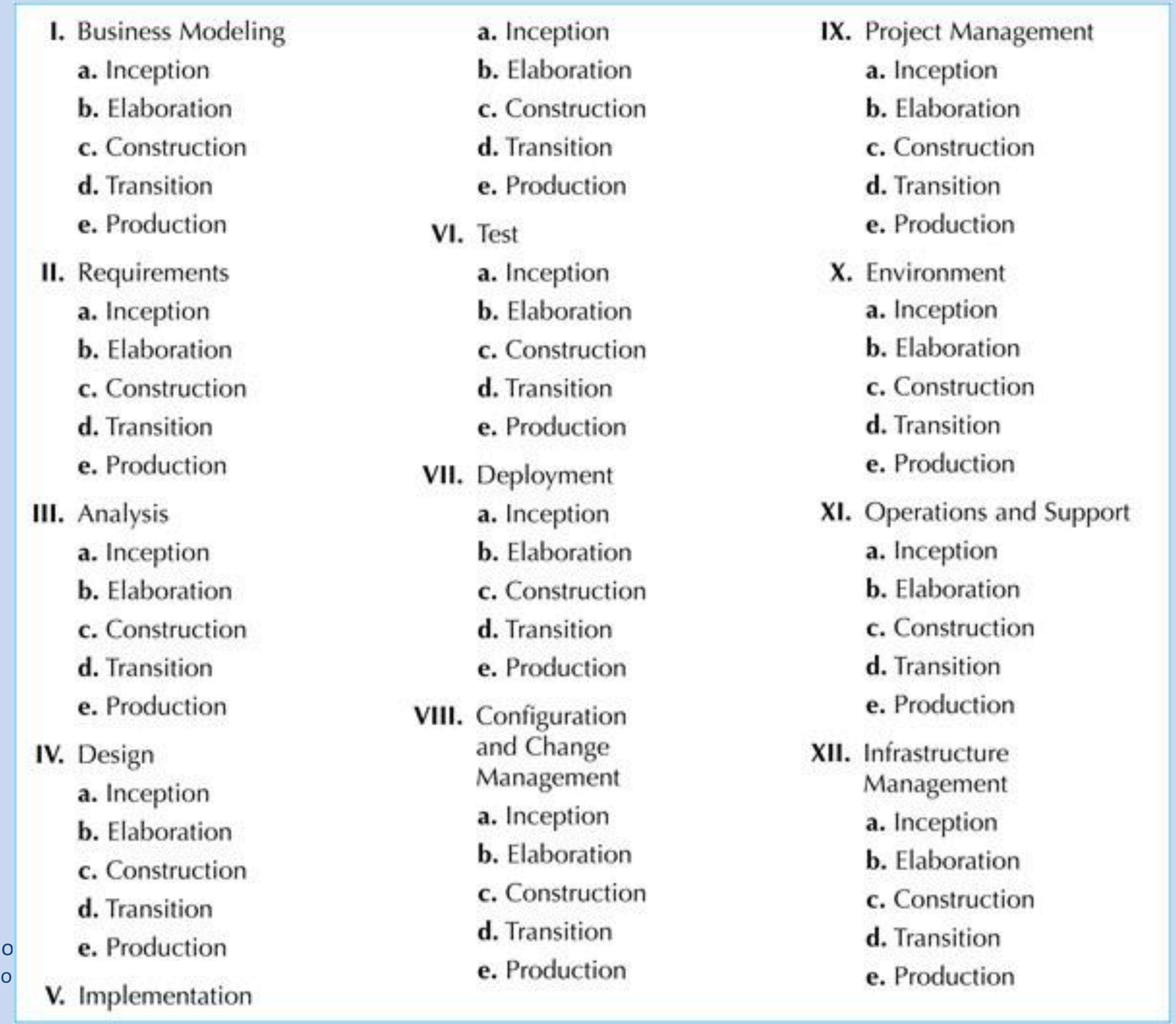
Evolutionary Work Breakdown Structures and Iterative Workplans

- Are created in an incremental and iterative manner
  - Encourages a more realistic view of both cost and schedule estimation

Evolutionary Work Breakdown Structures and Iterative Workplans

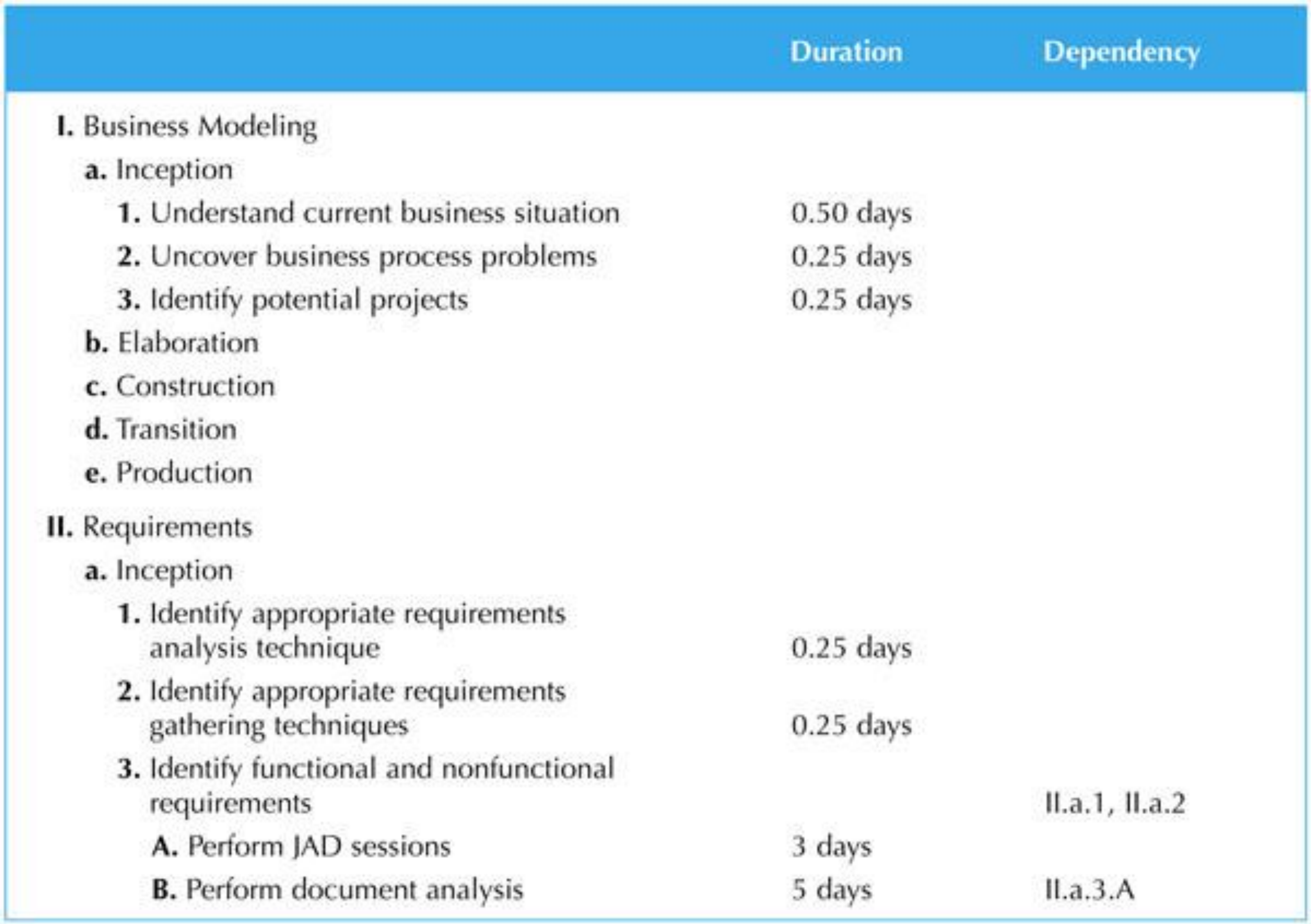
- Are not tied to a specific project
  - Enable the comparison of current project to earlier projects so learning from past successes and failures is possible

Evolutionary WBS for Enhanced Unified Process



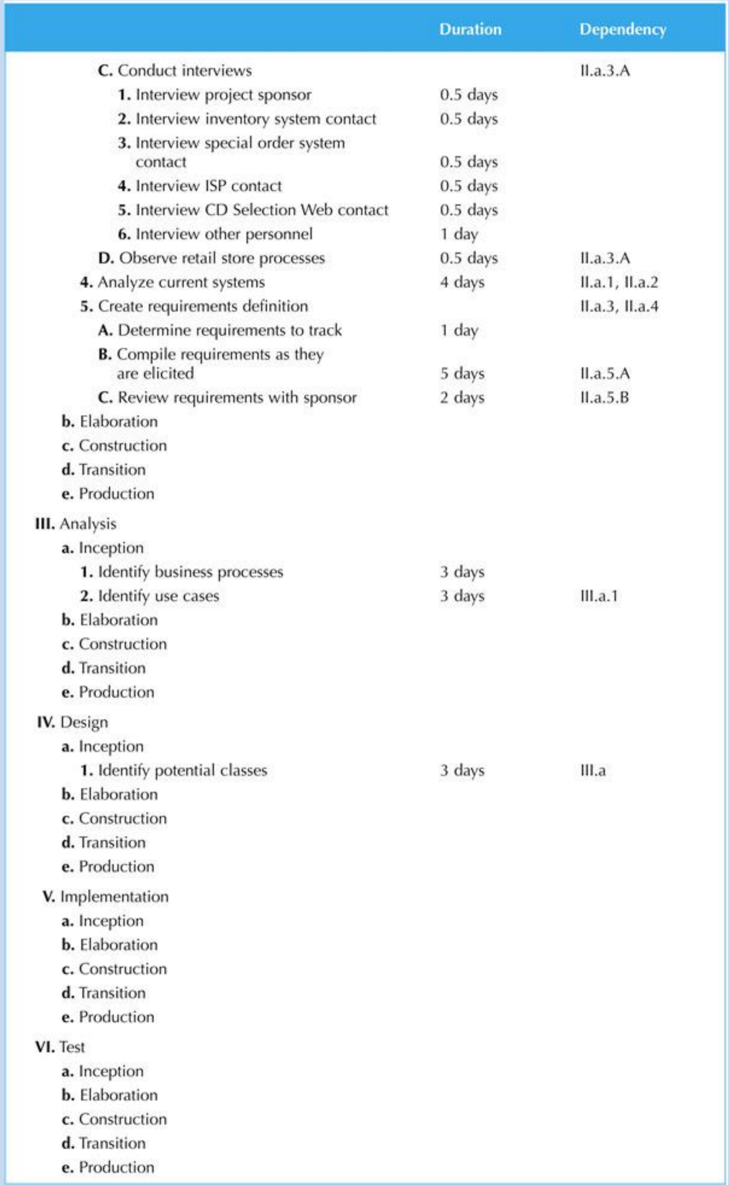
- **I.** Business Modeling
  - **a.** Inception
  - **b.** Elaboration
  - **c.** Construction
  - **d.** Transition
  - **e.** Production
- **II.** Requirements
  - **a.** Inception
  - **b.** Elaboration
  - **c.** Construction
  - **d.** Transition
  - **e.** Production
- **III.** Analysis
  - **a.** Inception
  - **b.** Elaboration
  - **c.** Construction
  - **d.** Transition
  - **e.** Production
- **IV.** Design
  - **a.** Inception
  - **b.** Elaboration
  - **c.** Construction
  - **d.** Transition
  - **e.** Production
- **V.** Implementation
  - **a.** Inception
  - **b.** Elaboration
  - **c.** Construction
  - **d.** Transition
  - **e.** Production
- **VI.** Test
  - **a.** Inception
  - **b.** Elaboration
  - **c.** Construction
  - **d.** Transition
  - **e.** Production
- **VII.** Deployment
  - **a.** Inception
  - **b.** Elaboration
  - **c.** Construction
  - **d.** Transition
  - **e.** Production
- **VIII.** Configuration and Change Management
  - **a.** Inception
  - **b.** Elaboration
  - **c.** Construction
  - **d.** Transition
  - **e.** Production
- **IX.** Project Management
  - **a.** Inception
  - **b.** Elaboration
  - **c.** Construction
  - **d.** Transition
  - **e.** Production
- **X.** Environment
  - **a.** Inception
  - **b.** Elaboration
  - **c.** Construction
  - **d.** Transition
  - **e.** Production
- **XI.** Operations and Support
  - **a.** Inception
  - **b.** Elaboration
  - **c.** Construction
  - **d.** Transition
  - **e.** Production
- **XII.** Infrastructure Management
  - **a.** Inception
  - **b.** Elaboration
  - **c.** Construction
  - **d.** Transition
  - **e.** Production

A Sample Evolutionary WBS for the planning of Inception Phase of Enhanced Unified Process

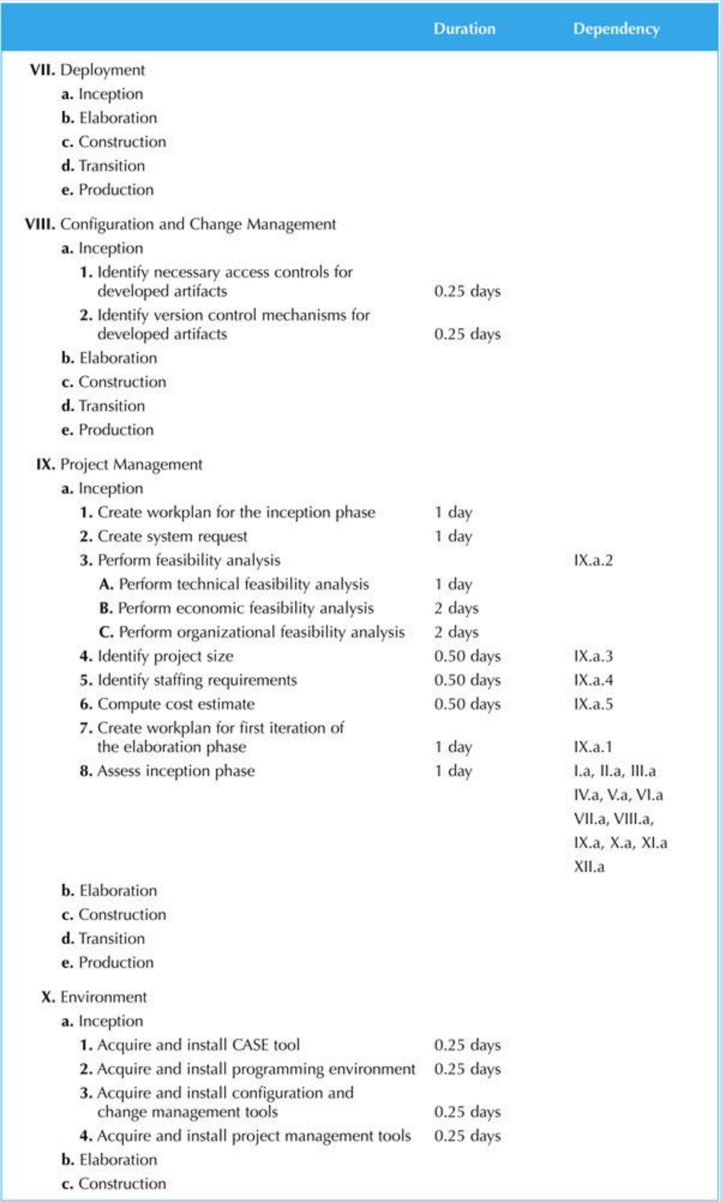


	Duration	Dependency
I. Business Modeling		
a. Inception		
1. Understand current business situation	0.50 days	
2. Uncover business process problems	0.25 days	
3. Identify potential projects	0.25 days	
b. Elaboration		
c. Construction		
d. Transition		
e. Production		
II. Requirements		
a. Inception		
1. Identify appropriate requirements analysis technique	0.25 days	
2. Identify appropriate requirements gathering techniques	0.25 days	
3. Identify functional and nonfunctional requirements		II.a.1, II.a.2
A. Perform JAD sessions	3 days	
B. Perform document analysis	5 days	II.a.3.A

A Sample Evolutionary WBS for the planning of Inception Phase of Enhanced Unified Process

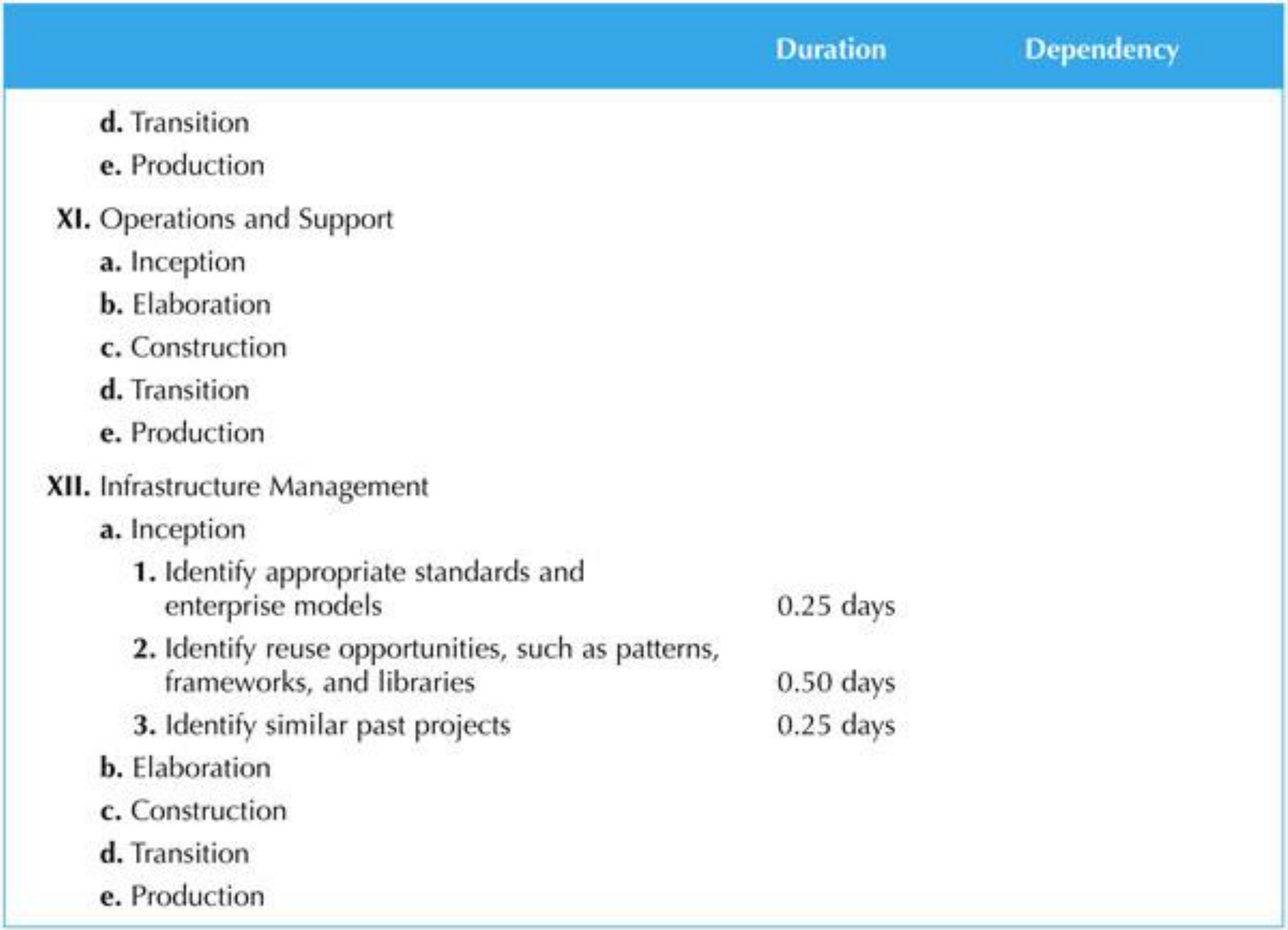


	Duration	Dependency
C. Conduct interviews		II.a.3.A
1. Interview project sponsor	0.5 days	
2. Interview inventory system contact	0.5 days	
3. Interview special order system contact	0.5 days	
4. Interview ISP contact	0.5 days	
5. Interview CD Selection Web contact	0.5 days	
6. Interview other personnel	1 day	
D. Observe retail store processes	0.5 days	II.a.3.A
4. Analyze current systems	4 days	II.a.1, II.a.2
5. Create requirements definition		II.a.3, II.a.4
A. Determine requirements to track	1 day	
B. Compile requirements as they are elicited	5 days	II.a.5.A
C. Review requirements with sponsor	2 days	II.a.5.B
b. Elaboration		
c. Construction		
d. Transition		
e. Production		
III. Analysis		
a. Inception		
1. Identify business processes	3 days	
2. Identify use cases	3 days	III.a.1
b. Elaboration		
c. Construction		
d. Transition		
e. Production		
IV. Design		
a. Inception		
1. Identify potential classes	3 days	III.a
b. Elaboration		
c. Construction		
d. Transition		
e. Production		
V. Implementation		
a. Inception		
b. Elaboration		
c. Construction		
d. Transition		
e. Production		
VI. Test		
a. Inception		
b. Elaboration		
c. Construction		
d. Transition		
e. Production		

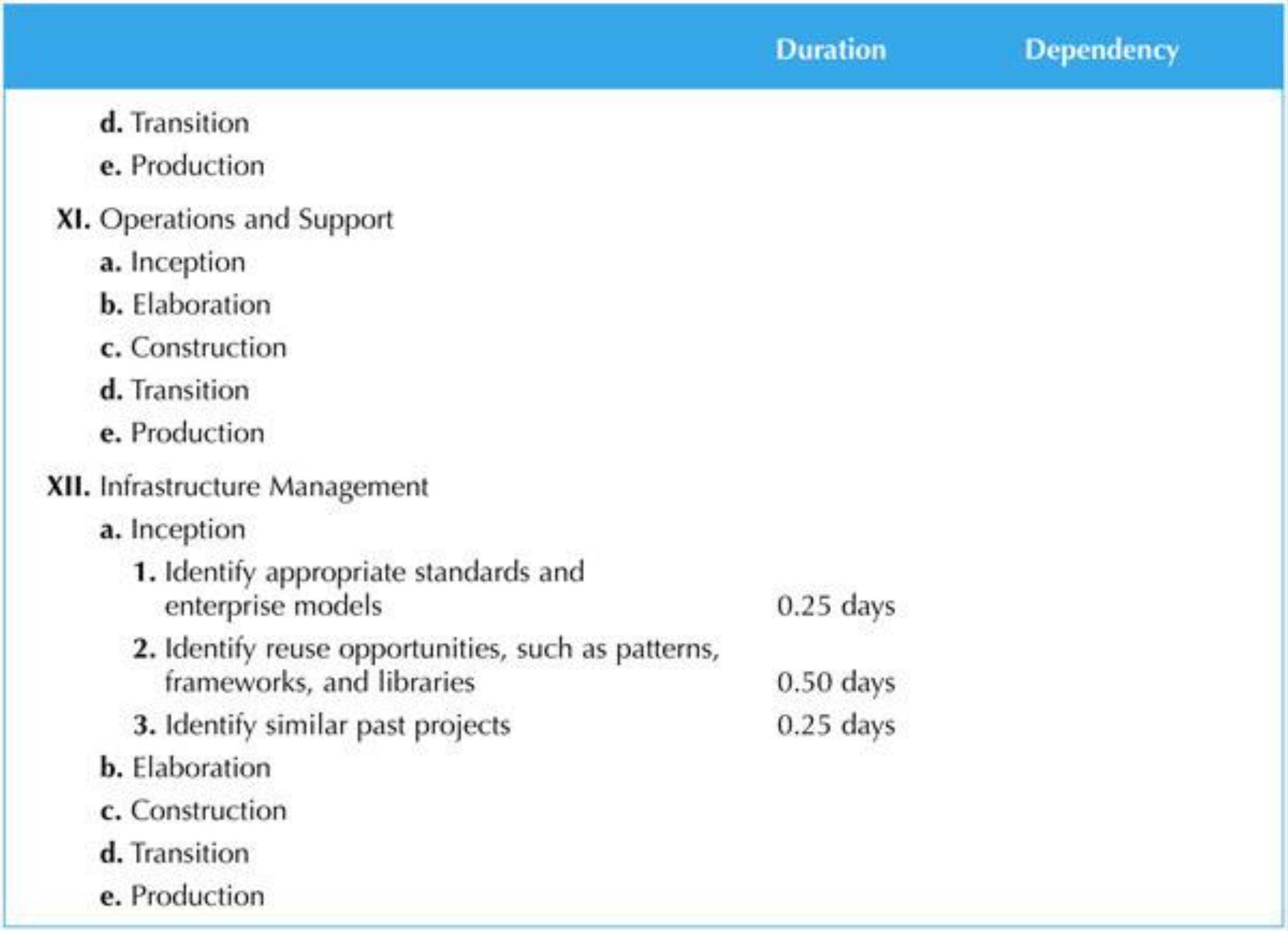


	Duration	Dependency
VII. Deployment		
a. Inception		
b. Elaboration		
c. Construction		
d. Transition		
e. Production		
VIII. Configuration and Change Management		
a. Inception		
1. Identify necessary access controls for developed artifacts	0.25 days	
2. Identify version control mechanisms for developed artifacts	0.25 days	
b. Elaboration		
c. Construction		
d. Transition		
e. Production		
IX. Project Management		
a. Inception		
1. Create workplan for the inception phase	1 day	
2. Create system request	1 day	
3. Perform feasibility analysis		IX.a.2
A. Perform technical feasibility analysis	1 day	
B. Perform economic feasibility analysis	2 days	
C. Perform organizational feasibility analysis	2 days	
4. Identify project size	0.50 days	IX.a.3
5. Identify staffing requirements	0.50 days	IX.a.4
6. Compute cost estimate	0.50 days	IX.a.5
7. Create workplan for first iteration of the elaboration phase	1 day	IX.a.1
8. Assess inception phase	1 day	I.a, II.a, III.a IV.a, V.a, VI.a VII.a, VIII.a, IX.a, X.a, XI.a XII.a
b. Elaboration		
c. Construction		
d. Transition		
e. Production		
X. Environment		
a. Inception		
1. Acquire and install CASE tool	0.25 days	
2. Acquire and install programming environment	0.25 days	
3. Acquire and install configuration and change management tools	0.25 days	
4. Acquire and install project management tools	0.25 days	
b. Elaboration		
c. Construction		

A Sample Evolutionary WBS for the planning of Inception Phase of Enhanced Unified Process



	Duration	Dependency
d. Transition		
e. Production		
XI. Operations and Support		
a. Inception		
b. Elaboration		
c. Construction		
d. Transition		
e. Production		
XII. Infrastructure Management		
a. Inception		
1. Identify appropriate standards and enterprise models	0.25 days	
2. Identify reuse opportunities, such as patterns, frameworks, and libraries	0.50 days	
3. Identify similar past projects	0.25 days	
b. Elaboration		
c. Construction		
d. Transition		
e. Production		



	Duration	Dependency
d. Transition		
e. Production		
XI. Operations and Support		
a. Inception		
b. Elaboration		
c. Construction		
d. Transition		
e. Production		
XII. Infrastructure Management		
a. Inception		
1. Identify appropriate standards and enterprise models	0.25 days	
2. Identify reuse opportunities, such as patterns, frameworks, and libraries	0.50 days	
3. Identify similar past projects	0.25 days	
b. Elaboration		
c. Construction		
d. Transition		
e. Production		

Staffing the Project

Staffing the Project

- Includes
  - How many people should be assigned
  - Matching people's skills with the needs of the project
  - Motivating them to meet the project objectives
  - Minimizing conflict that will occur over time

Staffing Plan

- Determine number and kind of people needed
- Overall reporting structure
- The project charter (describes the project's objectives and rules)

Creating a “Jelled” Team

- A team of people so strongly knit that the whole is greater than the sum of its parts
- Characteristics of a jelled team:
  - Very low turnover rate
  - Strong sense of identity
  - A feeling of eliteness
  - Team vs. individual ownership of the project
  - Team members enjoy their work

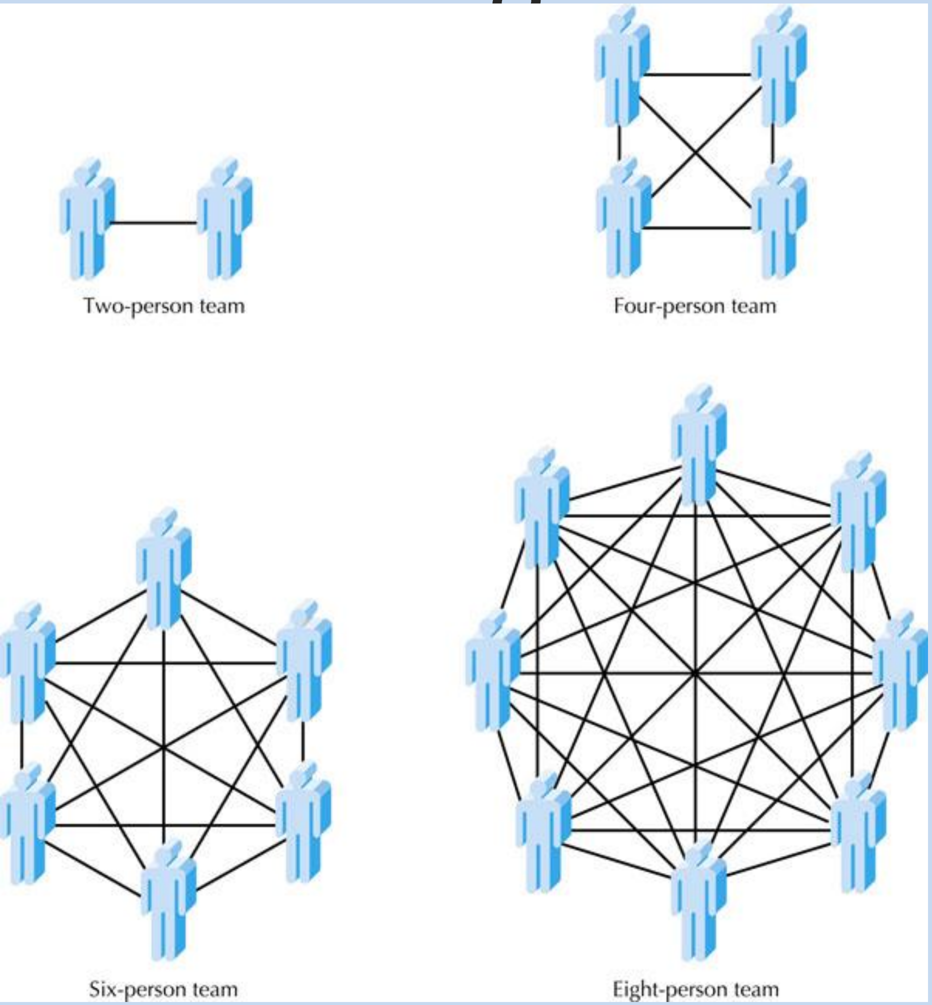
The Staffing Plan

- Calculate the number of people needed:

$$number\ of\ people = \frac{person\text{-}months}{time\ to\ complete\ (in\ months)}$$

- Lines of communication increase exponentially as people are added to a project

Staffing Plan



Increasing complexity with larger teams

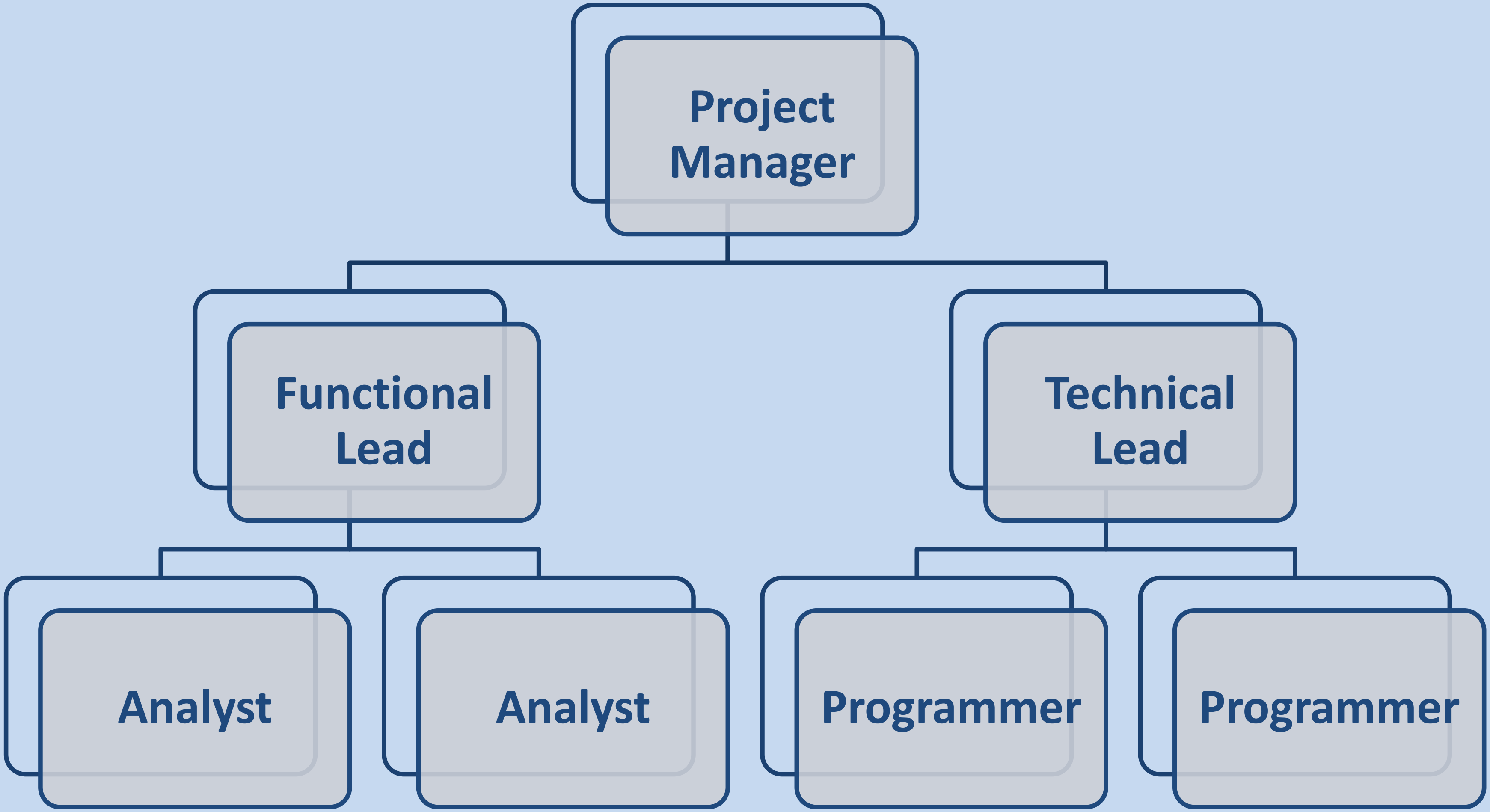
Staffing the Project

- To overcome this complexity build in a reporting structure that tempers its effects
- Rule of thumb is to keep team sizes fewer that 8-10; if more people are needed create subteams

Staffing the Project

- Decide on a reporting structure
- One person may fill more than one role
- Remember both technical and interpersonal skills are important in a project

Possible Reporting Structures



Motivation

- Motivation is the greatest influence on performance
- Use monetary rewards cautiously – generally last thing to do

Motivation

- Suggested motivating techniques:
  - 20% time rule
  - Peer-to-peer recognition awards
  - Team ownership (refer to the team as “we”)
  - Allow members to focus on what interests them
  - Utilize equitable compensation
  - Encourage group ownership
  - Provide for autonomy, but trust the team to deliver

Motivational Don’ts

- Assign unrealistic deadlines
- Ignore good efforts
- Create a low-quality product
- Give everyone on the project a raise
- Make an important decision without the team’s input
- Maintain poor working conditions

Conflict Avoidance Strategies

- Preventing or mitigating conflict:
  - Cohesiveness has the greatest effect
  - Clearly defining roles and holding team members accountable
  - Establish work & communications rules in the project charter

Conflict Avoidance Strategies

- Additional techniques:
  - Clearly define plans for the project
  - Make sure the team understands the importance of the project
  - Develop detailed operating procedures
  - Develop a project charter
  - Develop a schedule of commitments in advance
  - Forecast other priorities and their impact on the project

Standards

- One way to make certain that everyone is performing tasks in the same way and following same procedures is to create standards

Standards

- When a team forms standards and follows them, the project can be completed faster because task coordination becomes less complex

Standards

Types of Standards	Example
Documentation standards	The date and project name should appear as a header on all documentation.
Coding standards	All modules of code should include a header that lists the programmer, last date of update, and a short description of the purpose of the code.
Procedural standards	Report to project update meeting on Fridays at 3:30 PM. All changes to a requirements document must be approved by the project manager.
Specification requirement standards	Name of program to be created Description of the program's purpose
User interface design standards	The tab order of the screen will move from top left to bottom right. Accelerator keys will be provided for all updatable fields.

Environment & Infrastructure Management

- Environment—Choose the right set of tools
  - Use appropriate CASE tools to:
    - Increase productivity and centralize information (repository)
    - Utilize diagrams—more easily understood
  - Establish standards to reduce complexity

CASE Tools

- Not a silver bullet, but advantages include:
  - – Reduced maintenance costs
  - – Improve software quality
  - – Enforce discipline
  - – Some project teams even use CASE to assess the magnitude of changes to the project

Environment & Infrastructure Management

- Infrastructure—Document the project appropriately
  - Store deliverables & communications in a project binder
  - Use Unified Process standard documents
  - Don’t put off documentation to the last minute

Classical Planning Mistakes

1. Overly optimistic schedule
   - Solution : Do not inflate time estimates, instead explicitly schedule slack time at the end of each phase to account for variability in estimates

Classical Planning Mistakes

2. Failing to monitor schedule
   - Solution : Require team members to report progress honestly every week. There is no penalty for reporting a lack of progress but there are immediate sanctions for a misleading report

Classical Planning Mistakes

3. Failing to update the schedule
   - Solution : Immediately revise the schedule and inform project sponsor or use timeboxing to reduce functionality or move it into future versions

Classical Planning Mistakes

4. Adding people to a late project
   - Solution : Revise the schedule, use timeboxing, throw away bug-filled code, and add people only to work on an isolated part of the project

Summary

- Project Management
- Identifying Project Size
- Creating And Managing the Workplan
- Staffing the Project
- Coordinating Project Activities