System Analysis and Design

Eighth Edition

Alan Dennis, Barbara Wixom, Roberta M. Roth

Chapter 2

Project Selection and Management

Objectives

- Explain how projects are selected in some organizations.
- Describe various approaches to the systems development life cycle (SDLC) that can be used to structure a development project.
- Explain how to select a project methodology based on project characteristics.
- Describe project staffing issues and concerns.
- Describe and apply techniques to coordinate and manage the project.
- Explain how to manage risk on the project.

Introduction

- Most IT departments face a demand for IT projects that exceeds the department's ability to supply them
- Project portfolio management, a process of selecting, prioritizing, and monitoring project results, has become a critical success factor for IT departments facing too many potential projects
- Once selected, a systems development project undergoes a thorough process of project management
  - Project management is the process of planning and controlling the project within a specified time frame, at minimum cost, with the desired outcomes
- A project manager has the primary responsibility for managing the hundreds of tasks and roles that need to be carefully coordinated

Project Selection

- Many IT organizations tackle several important initiatives simultaneously
- These endeavors are managed as a program by the IT steering committee
- Investments in information systems projects today are evaluated in the context of an entire portfolio of projects
- Portfolio management takes into consideration the different kinds of projects that exist in an organization
- A good project portfolio will have the most appropriate mix of projects for the organization's needs
- The approval committee must be selective about where to allocate resources because the organization has limited funds

Ways to Classify Projects

Size	What is the size? How many people are needed to work on the project?
Cost	How much will the project cost the organization?
Purpose	What is the purpose of the project? Is it meant to improve the technical infrastructure? Support a current business strategy? Improve operations? Demonstrate a new innovation?
Length	How long will the project take before completion? How much time will go by before value is delivered to the business?
Risk	How likely is it that the project will succeed or fail?
Scope	How much of the organization is affected by the system? A department? A division? The entire corporation?
Economic Value	How much money does the organization expect to receive in return for the amount the project costs?

Project Selection Issues

- Project portfolio perspective – how does the project fit within the entire portfolio of projects
- Trade-offs needed: select projects to form a balanced project portfolio
- Viable projects may be rejected or deferred due to project portfolio issues

Creating the Project Plan

- Projects are launched after being selected by the approval committee
- The project manager then follows a set of project management guidelines as they organizes, guides, and directs the project from inception to completion
  - Sometimes referred to as the project management life cycle
- The project manager must make myriad decisions regarding the project, including determining the best project methodology, determining a staffing plan, and establishing mechanisms to coordinate and control the project

Project Methodology Options

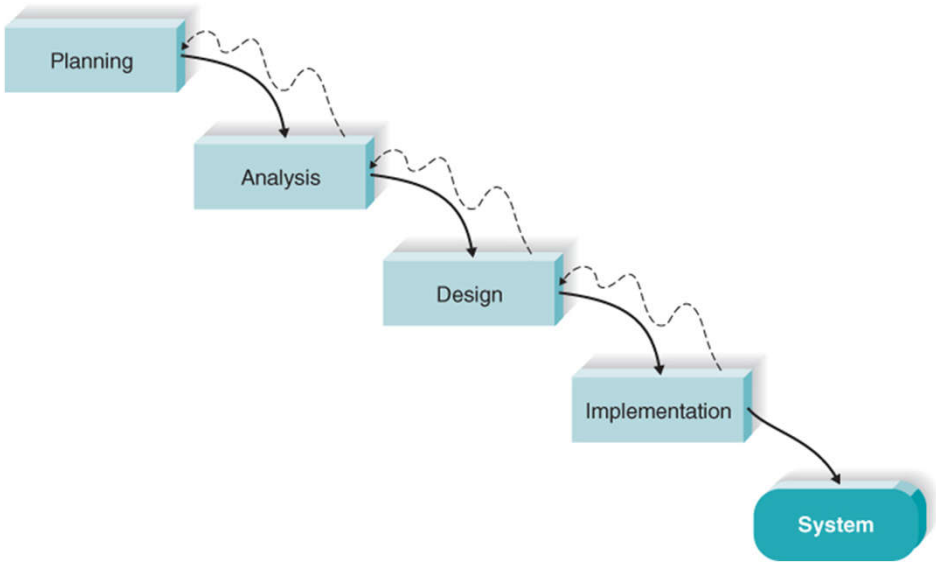
- The SDLC provides the foundation for the processes used to develop an information system
- A ***methodology*** is a formalized approach to implementing the SDLC
- There are many different systems development methodologies, and they vary in terms of the progression that is followed through the phases of the SDLC

Project Characteristics that will Affect the Methodology Selection Decision

Characteristic	Description
Clarity of User Requirements	How well do the users and analysts understand the functions and capabilities needed from the new system?
Familiarity with Technology	How much experience does the project team have with the technology that will be used?
System Complexity	How much complexity is anticipated in the new system? Does the new system include a wide array of features? Will the system have to integrate with many existing systems?
System Reliability	Will this system need to be highly reliable or is some downtime tolerable?
Short Time Schedules	Is the project time frame tight?
Schedule Visibility	Are the project sponsors, users, or organizational managers anxious to see progress?

Waterfall Development

- Move from phase to phase
- The key deliverables for each phase are typically voluminous
- Emphasis on deliverables from one phase flowing into the next phase



Waterfall Methodology Assessment

Strengths

- System requirements identified long before construction begins
- Requirements are “frozen” as project proceeds – no moving targets allowed

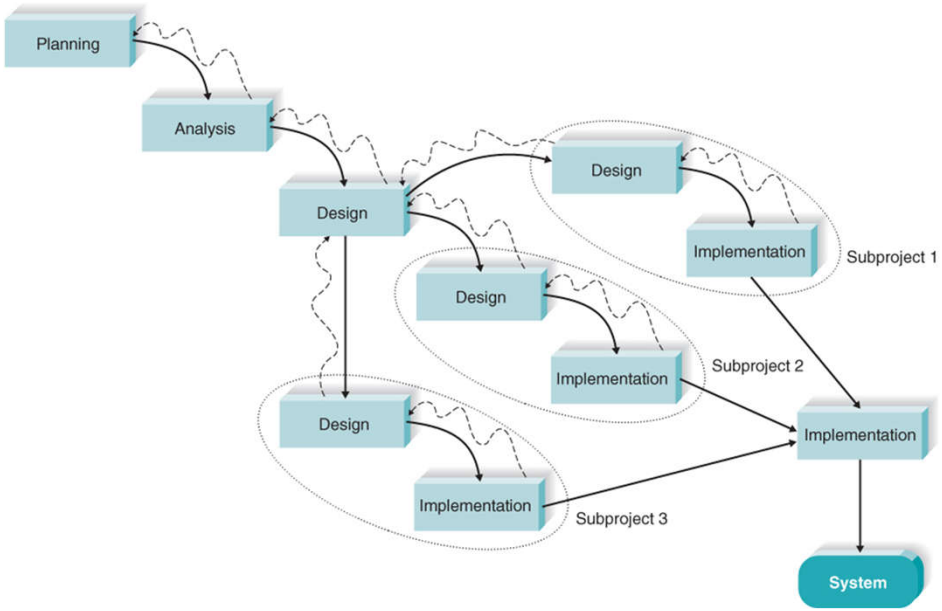
- **Weaknesses**
- Must wait a long time before there is “visible” evidence of the new system
- Takes a long time from start to finish

Several Important Variants of Waterfall Development

- Parallel development
- V-model

Parallel Development Methodology

- Subdivide the project into subprojects that can be worked on at the same time
- Reduce the overall project length



Parallel Methodology Assessment

Strengths

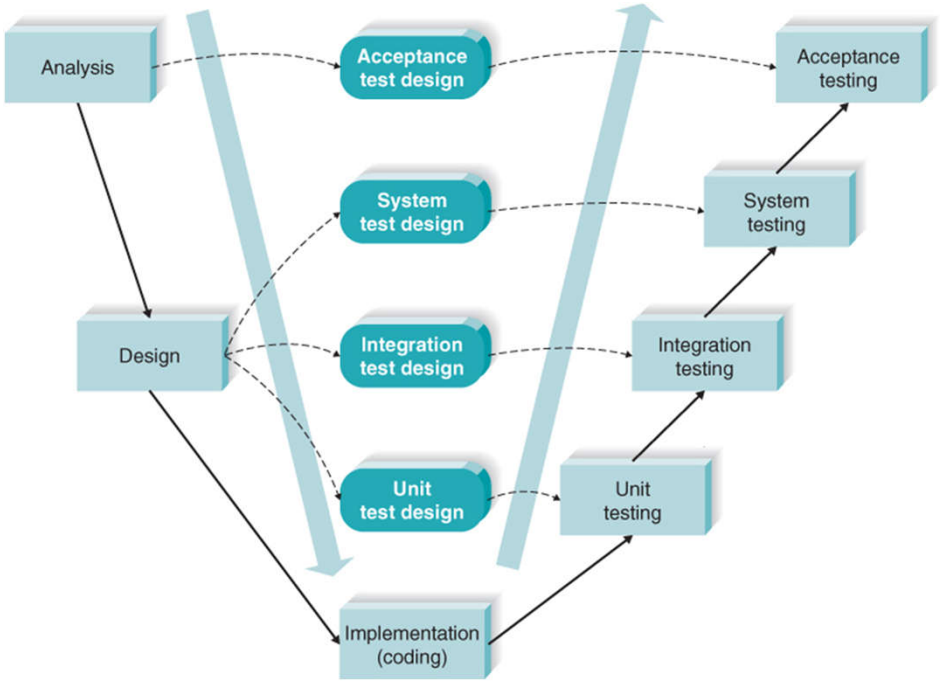
- Reduces overall project time (compared to Waterfall)
- Reduces the need for rework; with shorter time frame, less chance of requirements changing

Weaknesses

- Creating subprojects requires careful design decisions
- Integrating subprojects at the end can be complex and difficult

V-Model Development Methodology

- Emphasizes system quality through text plan development
- Simple and straightforward and improves the overall quality of systems through its emphasis on early development of test plans



V-Model Methodology Assessment

Strengths

- Simple and straightforward
- Quality improves through the emphasis on testing
- Including Quality Assurance expertise early in the project strengthens system quality

Weaknesses

- Rigid
- Difficult to use in a dynamic business environment

Rapid Application Development (RAD)

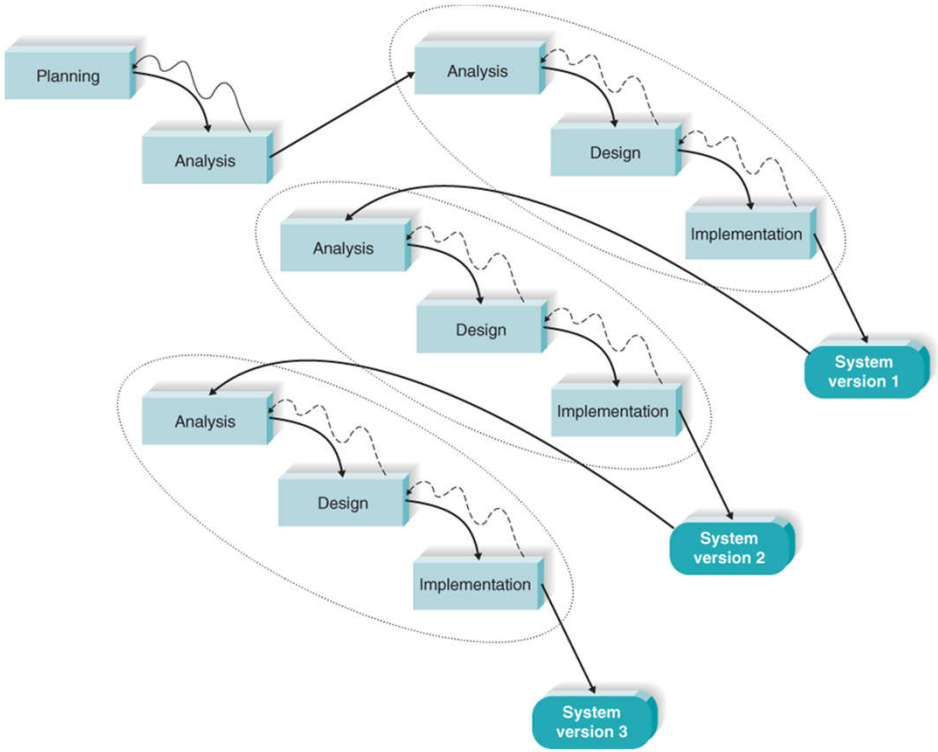
- Incorporates special techniques and tools
  - C A S E tools
  - J A D sessions
  - Visual programming languages
  - Code generators
- Goal – get some portion of system developed quickly and into the users' hands

Three R A D Approaches

1. Iterative development
   - A series of versions developed sequentially
2. System Prototyping
   - Create prototype (model) of system and “grow” it into the final system
3. Throw-away prototyping
   - Prototype alternative designs in an experimental way
   - Build system following prototype design but discard the actual prototype

Iterative Development Methodology

- Breaks the overall project into a series of versions that are developed sequentially
- Develop system in series of versions



Iterative Development Methodology Assessment

Strengths

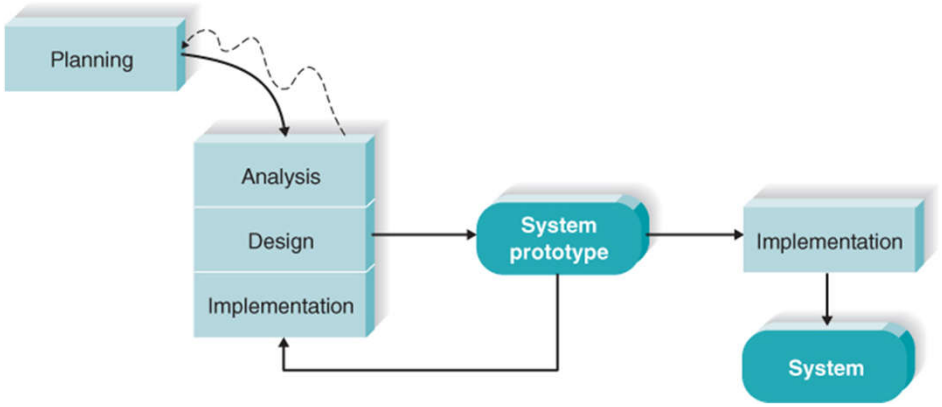
- Users get a system to use quickly
- Users identify additional needs for later versions based on real experiences with current version

Weaknesses

- Users faced with using an incomplete system for a time
- Users must be patient and wait for fully-functional system

System Prototyping Development Methodology

- Create a rough version of system quickly
- Following reaction and comments from the users, the developers reanalyze, redesign, and reimplement a second prototype
- Cycle continues until everyone agrees



System Prototyping Methodology Assessment

Strengths

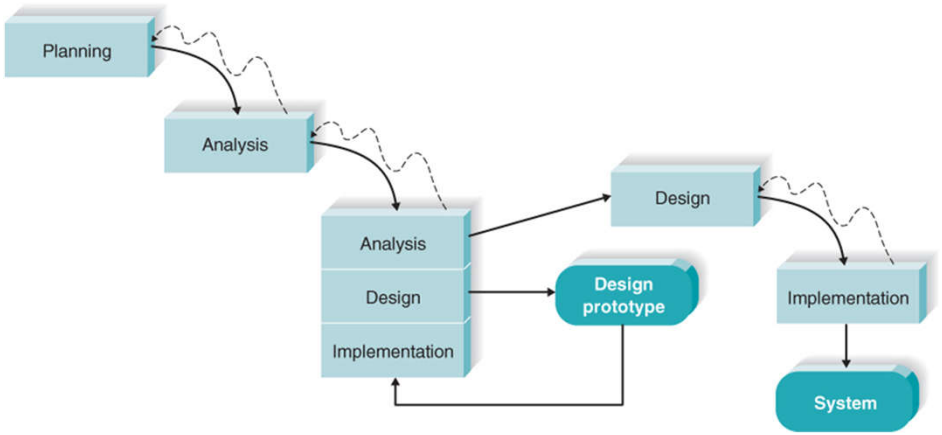
- Users get to work with prototype very quickly
- Feedback cycles let users identify changes and refine real requirements

Weaknesses

- Superficial analysis may cause problems
- Initial design decisions may be poor
- Overlooked features may be hard to add later

Throwaway Prototyping Development Methodology

- Adds emphasis on experimenting with design options before design is finalized
- Design options are thrown-away, but learning from them is factored into final design



Throwaway Prototyping Methodology Assessment

Strengths

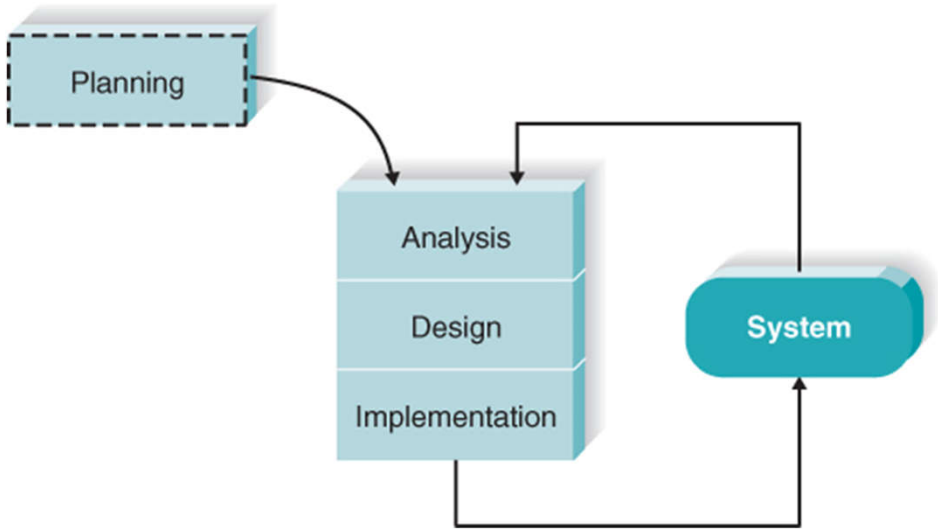
- Uncertainty is minimized
- Important issues are understood before building the final system

Weakness

- May take longer (compared to system prototyping)

Agile Development Methodology

- Extreme Programming (XP), Scrum, and others
- Focus on short cycles that produce a complete software product
- Highly adaptable in dynamic environments



Agile Methodologies Assessment

Strengths

- Fast delivery of results
- Works well in projects with undefined or changing requirements

Weaknesses

- Requires discipline
- Significant user involvement is essential
- Initial high learning curve
- Works best in smaller projects

Agile Versus Waterfall-Based Methodologies

- Agile development approaches have existed for several decades
- Created in part because of dissatisfaction with the sequential, inflexible structure of waterfall-based approaches
- Presently, agile development has made inroads into software development organizations, and studies show an even split between agile and waterfall users

Criteria for Selecting a Methodology

Usefulness in Developing Systems	Waterfall	Parallel	V-Model	Iterative	System Prototyping	Throwaway Prototyping	Agile Development
With unclear user requirements	Poor	Poor	Poor	Good	Excellent	Excellent	Excellent
With unfamiliar technology	Poor	Poor	Poor	Good	Poor	Excellent	Poor
That are complex	Good	Good	Good	Good	Poor	Excellent	Poor
That are reliable	Good	Good	Excellent	Good	Poor	Excellent	Good
With short time schedule	Poor	Good	Poor	Excellent	Excellent	Good	Excellent
With schedule visibility	Poor	Poor	Poor	Excellent	Excellent	Good	Good

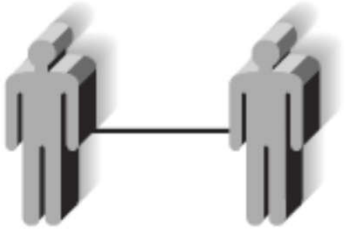
Staffing Plan

- The first step to staffing is determining the average number of staff needed for the project
- Divide the total person-months of effort by the optimal schedule
- The more a team grows, the more difficult it becomes to manage

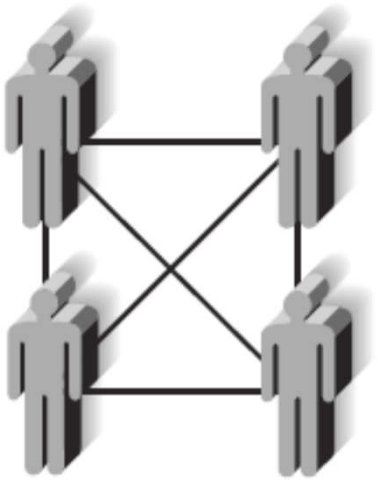
Staffing Considerations

- Match skills to project needs whenever possible
- Consider technical skills and interpersonal skills
  - All IS work is done in teams
  - Technical skills are not sufficient – need to be able to work with others
  - Use training and outside sources (consultants, vendor support) when skills are not readily available
- Staffing levels will change over a project's lifetime
- Adding staff adds overhead; not always productive

Increasing Complexity with Larger Teams

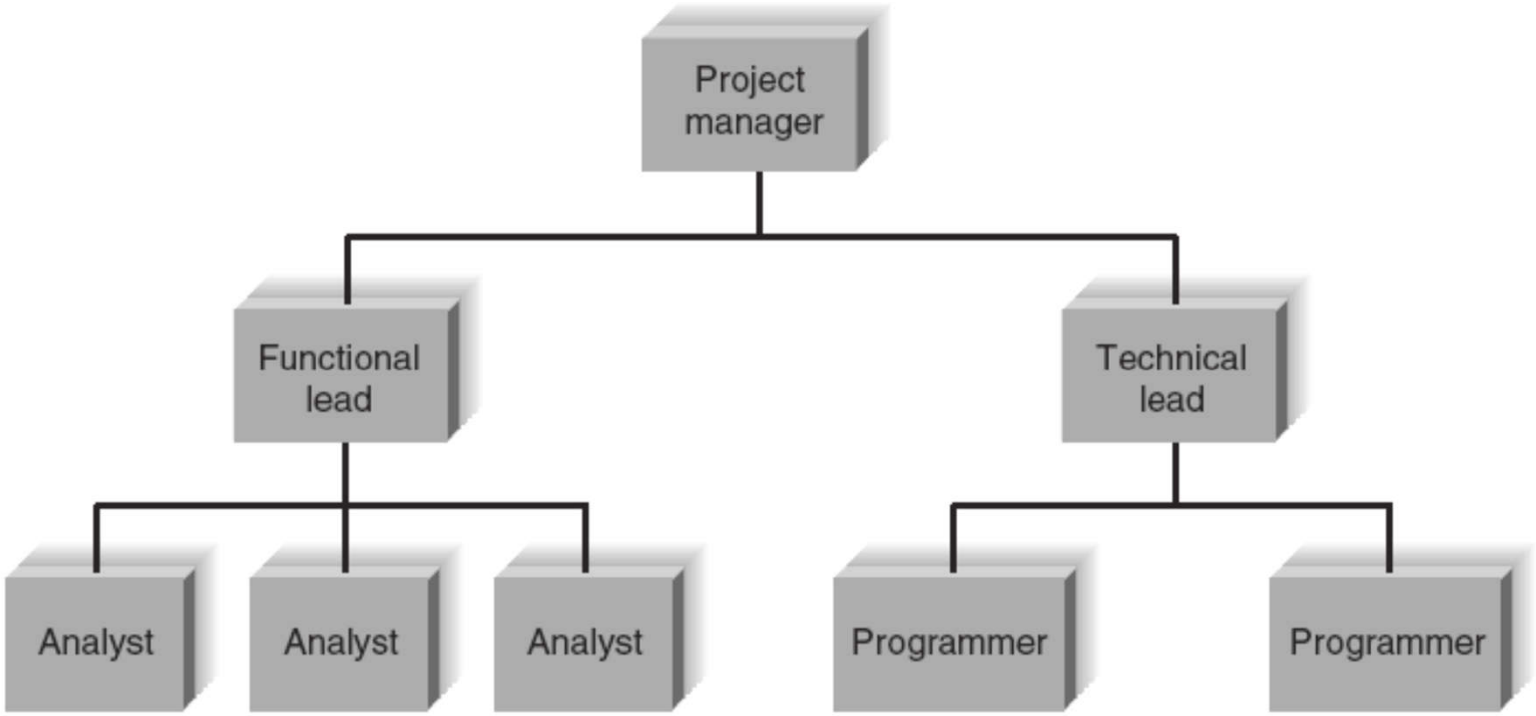


Two-person team



Four-person team

Possible Reporting Structure



Motivation

- Use monetary rewards cautiously
- Use intrinsic rewards
  - Recognition
  - Achievement
  - The work itself
  - Responsibility
  - Advancement
  - Chance to learn new skills

Motivational Don'ts

Don'ts	Reasons
Assign unrealistic deadlines	Few people will work hard if they realize that a deadline is impossible to meet.
Ignore good efforts	People will work harder if they feel that their work is appreciated. Often, all it takes is public praise for a job well done.
Create a low-quality product	Few people can be proud of working on a project that is of low quality.
Give everyone on the project a raise	If everyone is given the same reward, then high-quality people will believe that mediocrity is rewarded—and they will resent it.
Make an important decision without the team's input	Buy-in is particularly important. If the project manager needs to make a decision that greatly affects the members of her team, she should involve them in the decision-making process.
Maintain poor working conditions	A project team needs a good working environment, or motivation will go down the tubes. This includes lighting, desk space, technology, privacy from interruptions, and reference resources.

Handling Conflict

- Group cohesiveness is the attraction that members feel to the group and to other members of the group
- It contributes more to productivity than do project members' individual capabilities or experiences
- Defining the roles on the project and holding team members accountable for their tasks is a good way to begin mitigating potential conflict on a project
- Project charters can help

Conflict Avoidance Strategies

- Clearly define plans for the project
- Make sure the team understands how the project is important to the organization
- Develop detailed operating procedures and communicate these to the team members
- Develop a project charter
- Develop schedule commitments ahead of time
- Forecast other priorities and their possible impact on the project

Coordinating Project Activities

- The act of coordinating project activities continues throughout the entire project until a system is delivered
- This step includes putting efficient development practices in place and mitigating risk

Computer- Aided Software Engineering Tools

- CASE is a category of software that automates all or part of the development process
- Some CASE software packages are primarily used during the analysis phase to create integrated diagrams of the system and to store information regarding the system components
  - Upper CASE
- Other CASE software packages are design- phase tools that create the diagrams and then generate code for database tables and system functionality
  - Lower CASE

CASE Benefits and Drawbacks

- Benefits…
  - Tasks are much faster to complete and alter
  - Development information is centralized
  - Information is illustrated through diagrams
  - CASE can reduce maintenance costs
  - Improve software quality and enforce discipline
- Drawbacks
  - The advanced CASE tools are complex applications that require significant training and experience to achieve real benefits
  - Often, CASE serves only as a glorified diagramming tool

CASE Repository

- The central component of any CASE tool is the CASE repository
  - Also known as the information repository or data dictionary
- Stores the diagrams and other project information
- As the project evolves, project team members perform their tasks by using CASE and have access to each project team member's additions to the CASE repository

Standards

- Standards are created to ensure that team members are performing tasks in the same way and following the same procedures
- Standards can range from formal rules for naming files to forms that must be completed when goals are reached to programming guidelines
- Standards work best when they are created at the beginning of the project and well communicated to the entire project team

A Sampling of Project Standards

Types of Standards	Examples
Documentation	• The date and project name should appear as a header on all documentation. • All margins should be set to 1 inch. • All deliverables should be added to the project folder and recorded in its table of contents.
Coding	• 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. • Indentation should be used to indicate loops, if-then-else statements, and case statements. • On average, every program should include one line of comments for every five lines of code.
Procedural	• Record actual task progress in the work plan every Monday morning by 10 A.M. • Report to project update meeting on Fridays at 3:30 P.M. • All changes to a requirements document must be approved by the project manager.

A Sampling of Project Standards Continued

Types of Standards	Examples
Specifications requirement	• Name of the program to be created • Description of the program ' s purpose • Special calculations that need to be computed • Business rules that must be incorporated into the program • Pseudocode • Due date
User interface design	• Labels will appear in boldface text, left- justified, and followed by a colon. • The tab order of the screen will move from top left to bottom right. • Hot keys will be provided for all updatable fields.

Documentation

- Another technique that project teams put in place during the planning phase is good documentation
- Often, the documentation is stored on a file server available to the entire team
  - Project binder
- A simple way to set up your documentation is to create a folder hierarchy and use subfolders to separate content according to the major phases of the project

Managing and Controlling the Project

- The science of project management is in making trade-offs among three important concepts:
  1. The size of the system (in terms of what it does)
  2. The time to complete the project (when the project will be finished)
  3. The cost of the project
- The project manager must work with the project sponsor to shift these appropriately as the project progresses
- Once the project begins, the project manager monitors the progress of the team on the project tasks as the project team members make periodic status reports

Refining Estimates

- Even projects with high-quality estimates will need refinement
- Project managers must adjust estimated time throughout the project

Project Estimates Require Refinement

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

Managing Scope

- Beware of scope creep
- Implement formal change approval process
- Defer additional requirements as future system enhancements

Timeboxing

- Timeboxing sets a fixed deadline for a project and delivers the system by that deadline no matter what, even if functionality needs to be reduced
- Time estimating techniques may reveal that the project requires more time than we have available
- Timeboxing helps in these situations
  - Set a tight but realistic deadline. Identify core, essential functional requirements
  - Team limits its focus just to essential functions
  - High quality is stressed
  - Other functions will be added later
  - Repeat to add refinements and enhancements

Steps for Timeboxing

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

Managing Risk

- ***Risk management*** is the process of assessing and addressing the risks that are associated with developing a project
- Typically, project teams create a risk assessment, or a document that tracks potential risks along with an evaluation of the likelihood of the risk and its potential impact on the project
- Most project managers keep abreast of potential risks, even prioritizing them according to their magnitude and importance
- Over time, the list of risks will change

Chapter Review

- Explain how the practice of project portfolio management may influence the selection of IS projects.
- Discuss the skills needed to be a successful systems analyst.
- List and explain the project characteristics that affect the selection of a project methodology.
- List and explain three methodologies that are based on the waterfall concept.
- List and explain three methodologies that are based on RAD.
- Explain the XP Agile methodology.

Chapter Review Continued

- For each methodology included in the chapter, summarize the project characteristics that make that methodology the best choice and the poorest choice. Explain why.
- Discuss the three main tasks involved when staffing a project.
- Describe various ways to influence the motivation of project team members.
- Explain the purpose and content of a project charter.
- Describe the role of CASE tools in coordinating the project.
- Describe the value of standards to the project team.

Chapter Review Continued

- Discuss the project manager's balancing act involving size, time, and cost.
- Describe how scope creep affects a project.
- Discuss the technique of timeboxing and how it affects a project team.

Key Terms

- Agile development
- CASE repository
- Chief information
- officers (CIOs)
- Computer-aided
- software
- engineering (CASE)
- Design prototype
- Documentation
- Feature creep
- Functional lead
- Group cohesiveness
- Integrated CASE
- Interpersonal skills
- Iterative development
- Lower CASE
- Methodology
- Motivation
- Parallel development
- Project binder
- Project management
- Project manager
- Project portfolio
- management
- Rapid application
- development (RAD)
- Reporting structure
- Risk assessment
- Risk management
- Scope creep
- Staffing plan
- Standards
- System prototyping
- Technical lead
- Technical skills
- Throwaway prototyping
- Timeboxing
- Trade-offs
- Upper CASE
- Versions
- V-model
- Waterfall development