Chapter 2

Project Selection and Management

SYSTEMS ANALYSIS AND DESIGN

SEVENTH EDITION

DENNIS, WIXOM, AND ROTH

Learning Objectives

- Explain how projects are selected in some organizations.
- Describe various approaches to the SDLC that can be used to structure a development project.
- Explain how to select a project methodology based on project characteristics.
- Become familiar with project estimation.
- Be able to create a project work plan.
- Describe project staffing issues and concerns.
- Describe and apply techniques to coordinate and manage the project.
- Explain how to manage risk on the project.

Project Selection

HOW SPECIFIC PROJECTS ARE CHOSEN

Project Selection Issues (1 of 2)

- Ways to Characterize Projects
  - Size
  - Cost
  - Purpose
  - Length
  - Risk
  - Scope
  - Economic Value

Project Selection Issues (2 of 2)

- Approval committee uses the system request and the feasibility study
  - 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.

Project Portfolio Management

- PPM software collects and manages information about all projects – on-going and awaiting approval.
- Companies stay up to date on projects and adapt to changing conditions.
- Features: project prioritization, employee allocation, real-time project monitoring, flagging cost and time variances, monitoring economic feasibility.

Creating the Project Plan

- Upon project approval, the project manager must:
  - Select the best project methodology
  - Develop a project work plan
  - Establish a staffing plan
  - Create ways to coordinate and control the project

Creating the Project Plan

DEVELOPING A PLAN FOR A SUCCESSFUL RESULT

Selecting a Project Methodology

- Methodology: A formalized approach to implementing the SDLC
  - A series of steps to perform and deliverables to produce
- Methodology Sources
  - Internally developed by organizations
  - Consulting firms
  - Software vendors
  - Government agencies

Selecting a Project Methodology - Issues

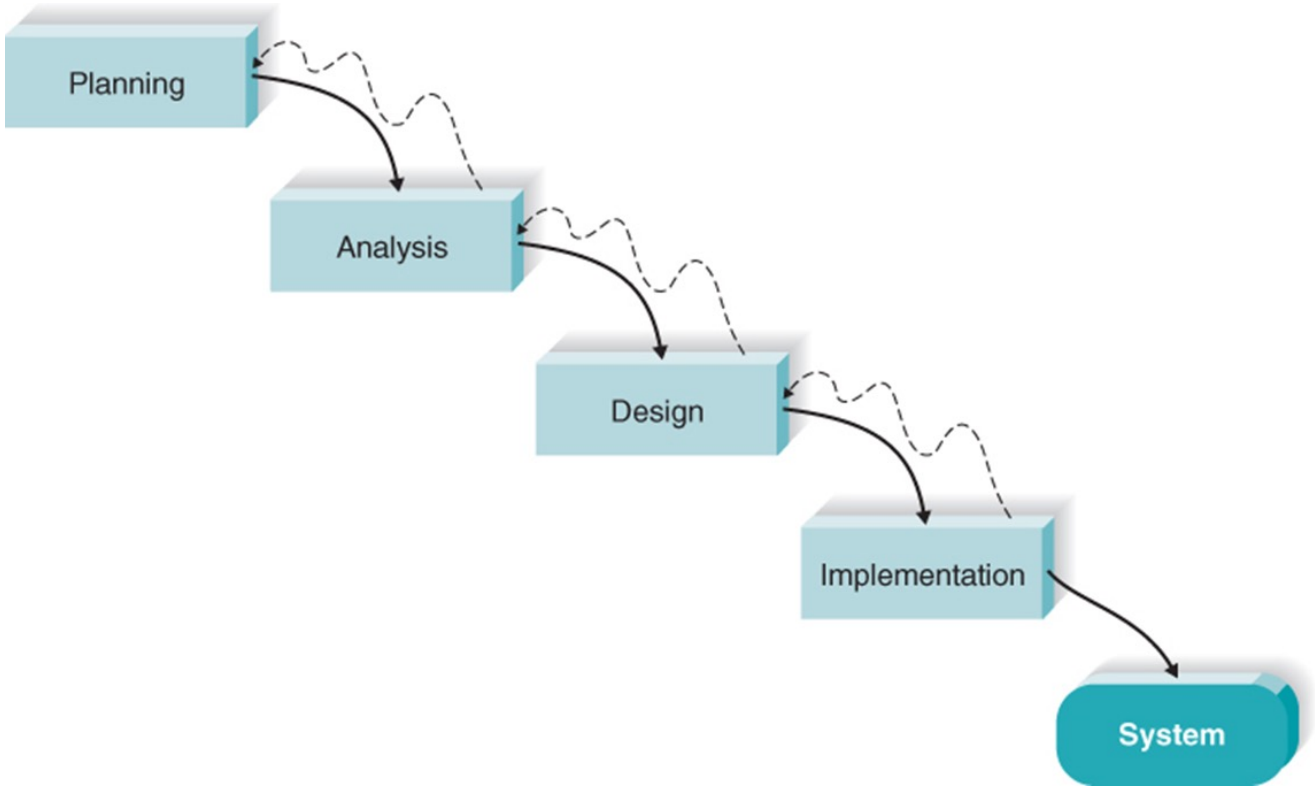
- These factors influence the best choice:
  - Clarity of User Requirements
  - Familiarity with Technology
  - System Complexity
  - System Reliability
  - Time Frame
  - Schedule Visibility

Structured Systems Development

- Based upon SDLC
- Assumes a project phase is complete before moving to the next phase
  - Waterfall Development
  - Parallel Development
  - V-model
- Goal – doing each phase thoroughly before moving forward ensures correct and high-quality outcomes

Waterfall Development Methodology

- Move from phase to phase
- Emphasis on deliverables from one phase flowing into the next phase



Waterfall Methodology Assessment

STRENGTHS

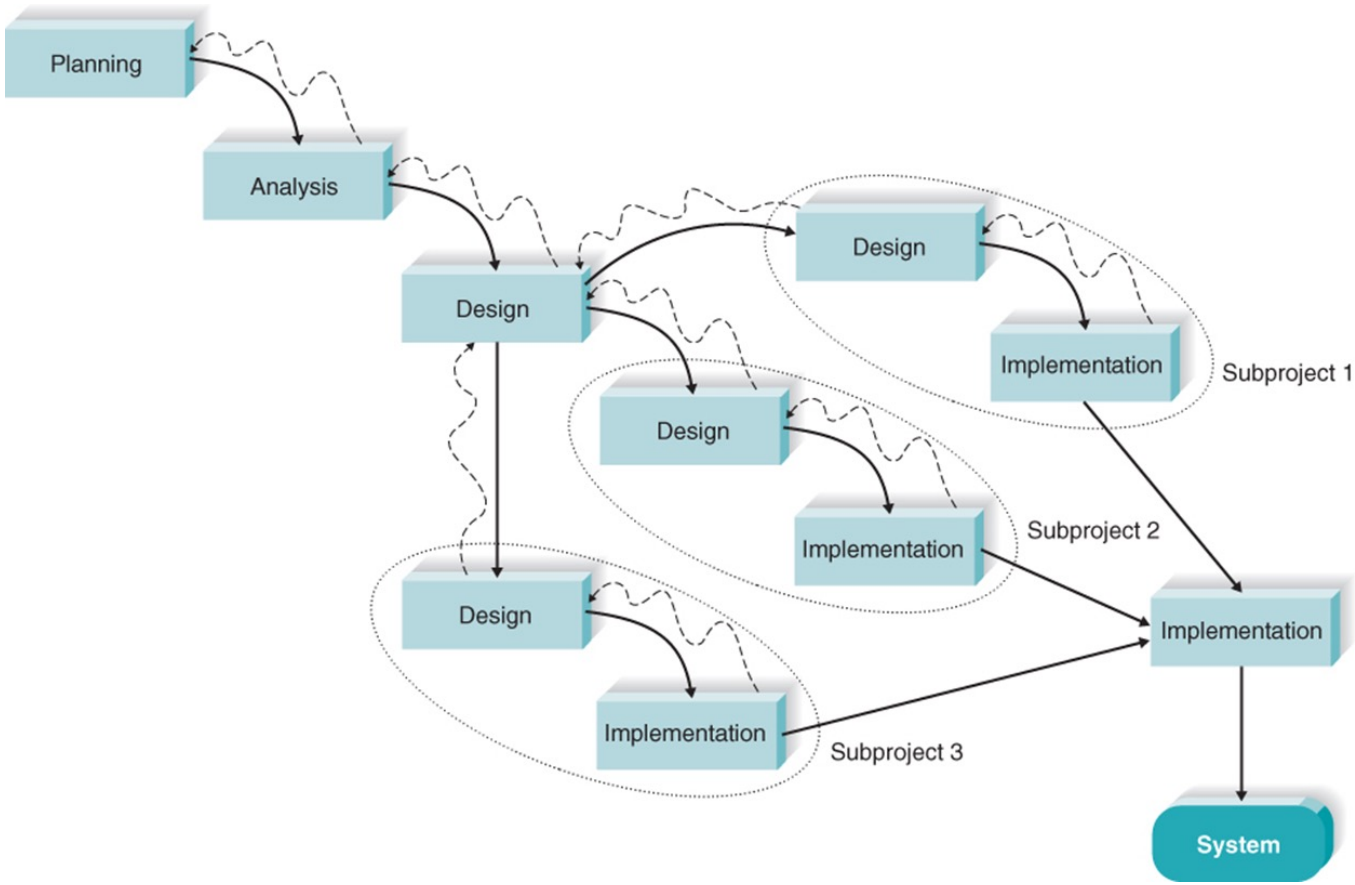
- System requirements identified long before construction begins
- Requirements are “frozen” as project proceeds – no moving targets allowed

WEAKNESSES

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

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

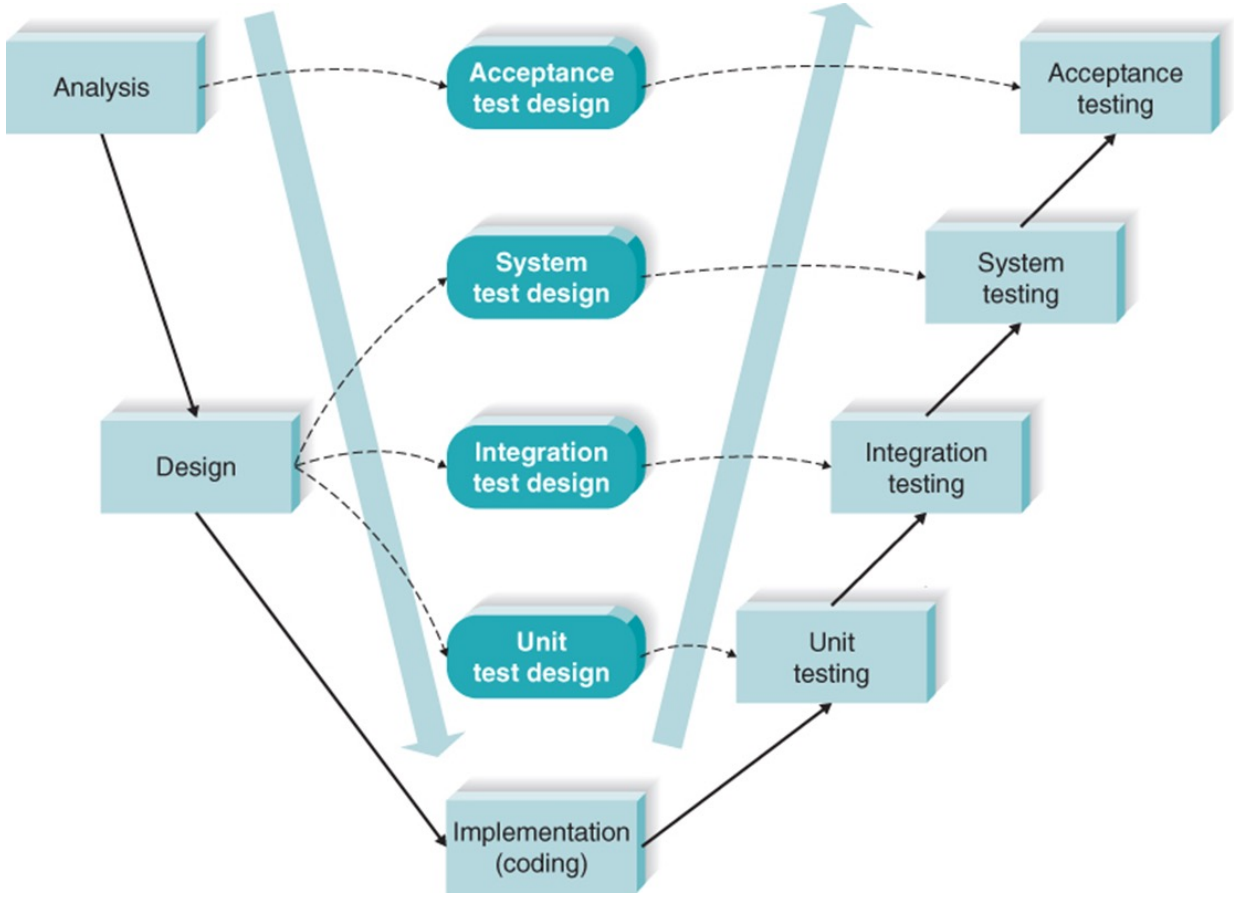
- Reduces overall project time (compared to Waterfall)
- Reduces the need for rework; with shorter time frame, less chance of requirements changing

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



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

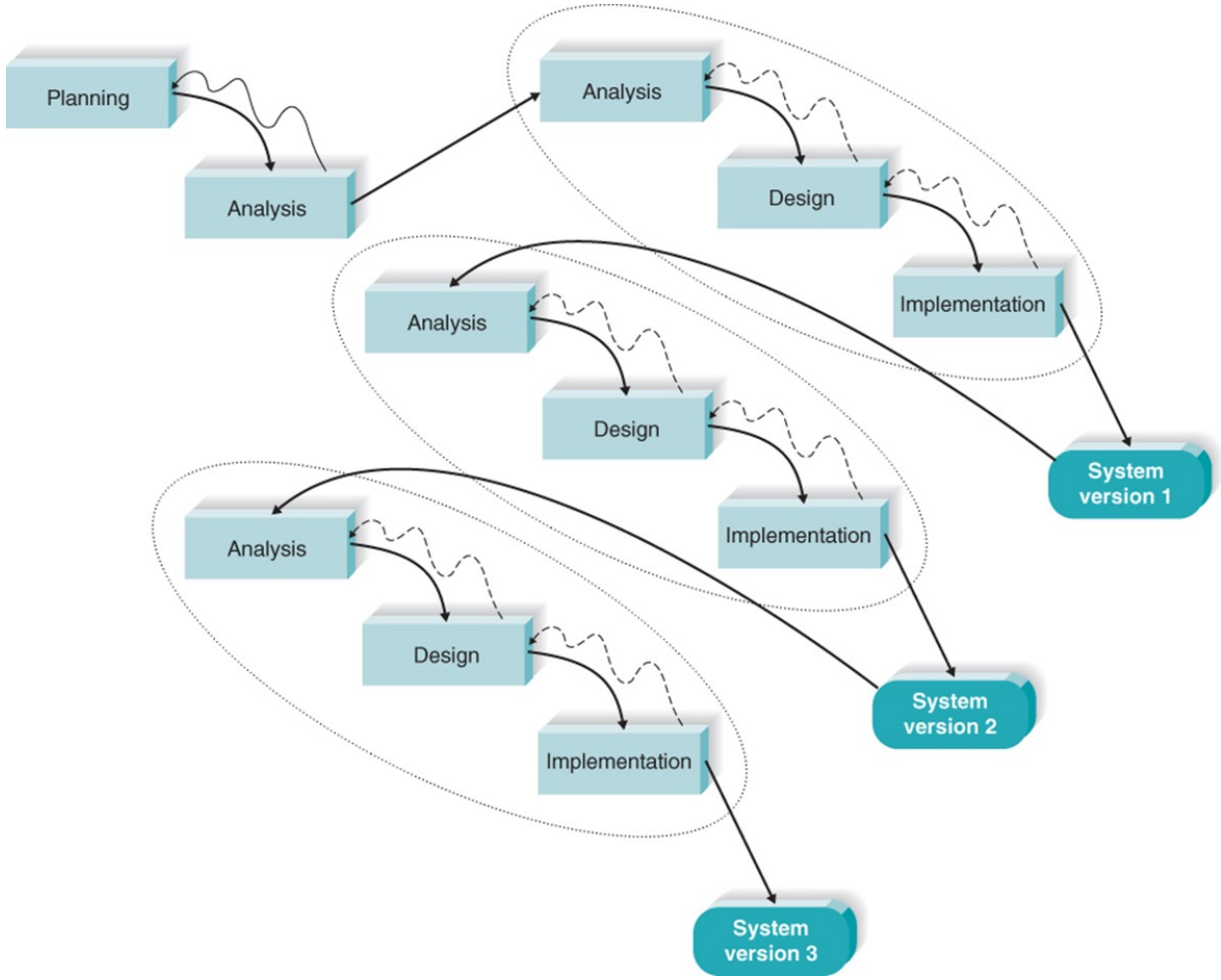
- Incorporates special techniques and tools
  - CASE tools
  - JAD sessions
  - Visual programming languages
  - Code generators
- Goal – get some portion of system developed quickly and into the users' hands

Three RAD Approaches

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

Iterative Development Methodology

- RAD approach
- Develop system in series of versions



Iterative Development Methodology Assessment

STRENGTHS

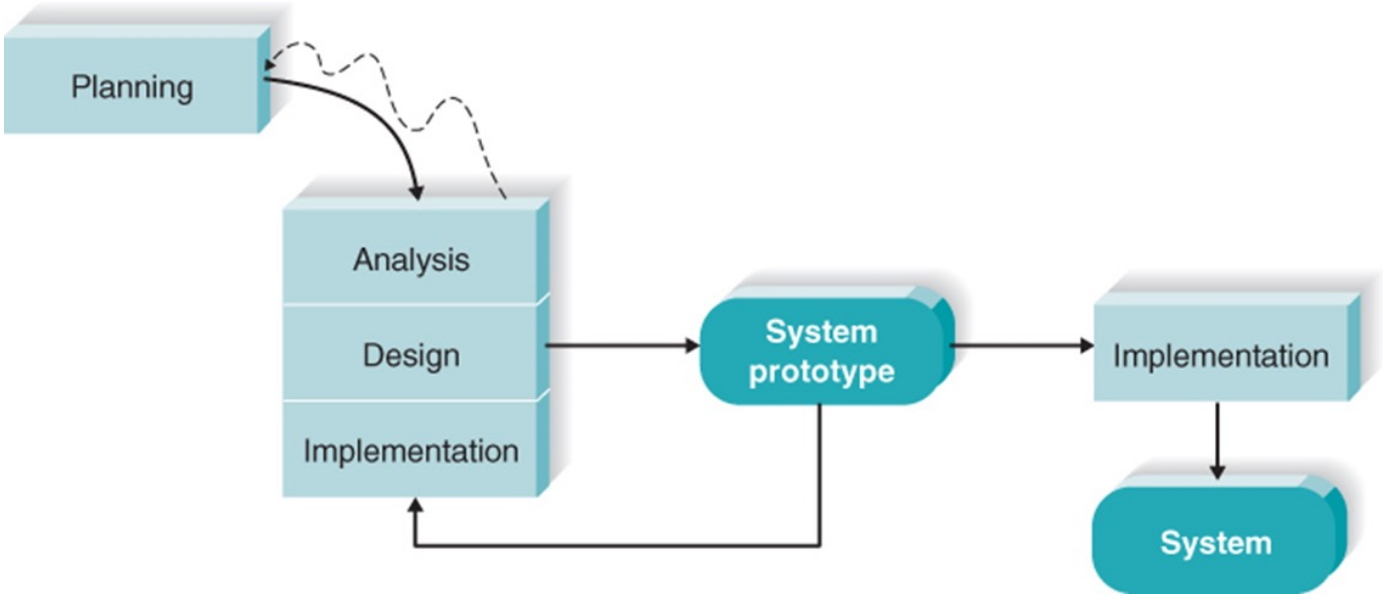
- Users get a system to use quickly
- Users identify additional needs for later versions based on real experiences with current version

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

- RAD approach
- Create a rough version of system quickly and "grow" it into final system with repetitive refinement



System Prototyping Methodology Assessment

STRENGTHS

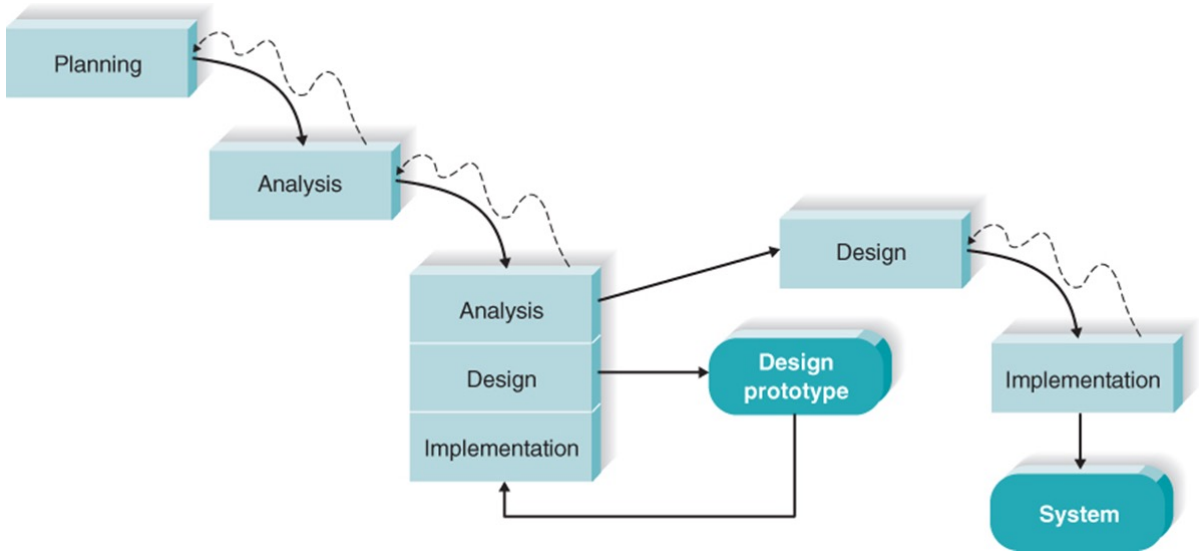
- Users get to work with prototype very quickly
- Feedback cycles let users identify changes and refine real requirements

WEAKNESSES

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

Throwaway Prototyping Development Methodology

- R A D approach
- 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

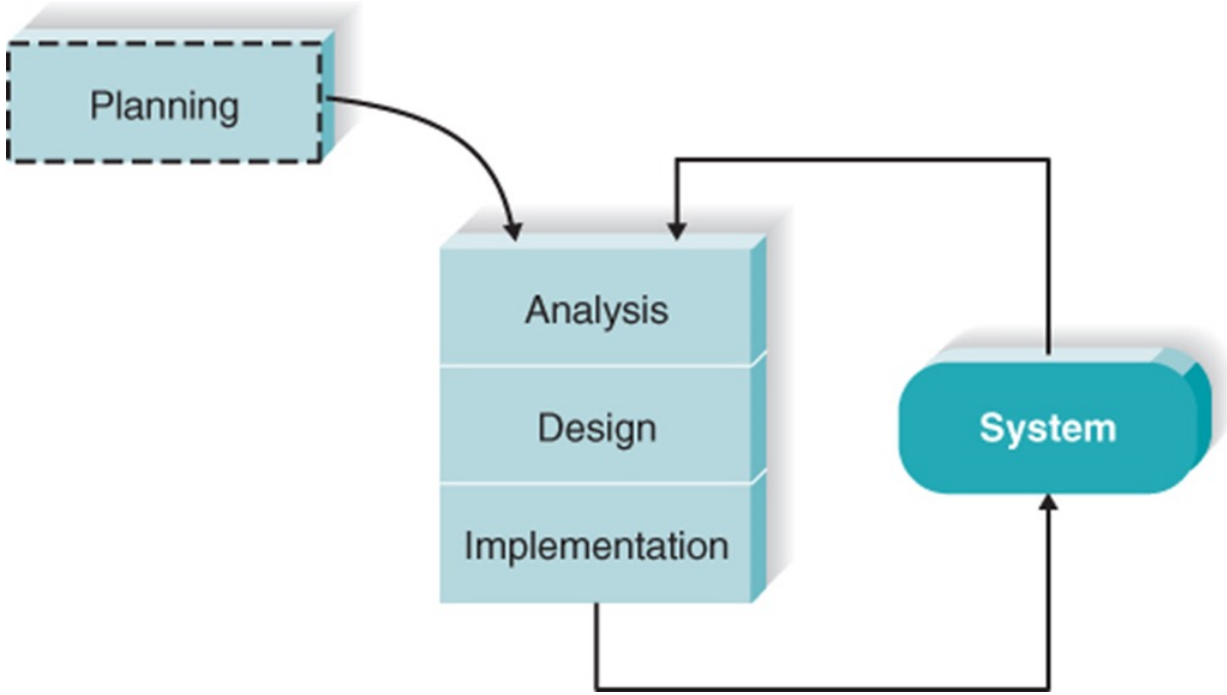
- Uncertainty is minimized
- Important issues are understood before building the final system

WEAKNESSES

- 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

Selection Summary

Ability to develop systems	Waterfall	Parallel	V-Model	Iterative	System Proto-typing	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 a short time schedule	Poor	Good	Poor	Excellent	Excellent	Good	Excellent
With schedule visibility	Poor	Poor	Poor	Excellent	Excellent	Good	Good

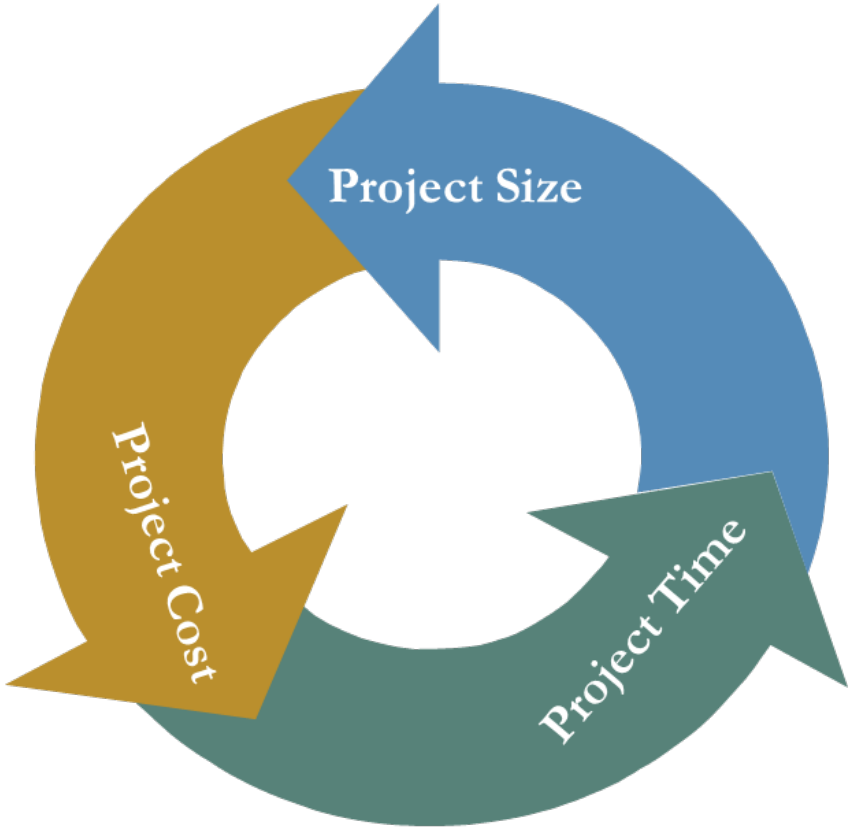
Project Management Tasks

PREPARING TO LAUNCH THE PROJECT

Project Manager's Balancing Act

Project Management involves making trade-offs…

Modifying one element requires adjusting the others



Project Estimation

- The process of assigning projected values for time and effort
- Sources of estimates
  - Methodology in use
  - Actual previous projects
  - Experienced developers
- Estimates begin as a range and become more specific as the project progresses
  - Industry standards
  - Function point estimation (Appendix 2A)

Project Estimates Using Industry Standard Percentages

INDUSTRY STANDARD PERCENTAGES

	Planning	Analysis	Design	Implementation
Typical industry standards for business applications (%)	15	20	35	30
Estimates based on actual figures for the first stage of SDLC* (person-months)	Actual: 4	Estimated: 5.33	Estimated: 9.33	Estimated: 8

*SDLC, systems development life cycle.

EXAMPLE

IF 4 months are required for Planning, then

15% X = 4, where X = overall length of project

$X = 4 / 15\%$

X = 26.66 months for entire project

Therefore:

- Planning (15%): 4 months
- Analysis (20%): 5.33 months
- Design (35%): 9.33 months
- Implementation (30%): 8 months

- Total Project Length: 26.66 months

Identifying Tasks

- Use established guidelines – existing methodologies
- Use analogies – model previous projects' task lists
- Top-down approach – break high level tasks into smaller, detailed tasks
- Organize into work breakdown structure

Example – Determining Tasks using Top-down Approach

- Grade programming assignments
  1. Create grading plan
     - A. Develop grading rubric
     - B. Develop test plan, test data, and check figures
  2. Prepare programming projects for grading
     - A. Download submitted projects
     - B. For all projects, extract zipped files
  3. For all projects,
     - A. Administer test plan
       - Check performance
       - Verify results
     - B. Check code for required elements and standards
     - C. Apply rubric and determine final score.

Typical Workplan Entry

Task Information	Example
Name of the task	Perform economic feasibility
Start date	Jan 5, 2019
Completion date	Jan 19, 2019
Person assigned to the task	Project sponsor Mary Smith
Deliverable(s)	Cost—benefit analysis
Completion status	Complete
Priority	High
Resources needed	Spreadsheet software
Estimated time	16 hours
Actual time	14.5 hours

Project Work Plan

			Estimated			Actual				
Task ID	Task Name	Assigned To	Duration (days)	Start Date	Finish Date	Start Date	Finish Date	Duration variance	Dependency	Status
1	Design Phase		31	Mon 1/6/20	Mon 2/17/20					Open
1.1	Develop database design document	Megan	9	Mon 1/6/20	Thurs 1/16/20					Open
1.1.1	Staging database design	Megan	9	Mon 1/6/20	Thurs 1/16/20					Open
1.1.2	Suspense database design	Megan	9	Mon 1/6/20	Thurs 1/16/20					Open
1.2	Develop rejects-handling design document	Megan	9	Fri 1/17/20	Wed 1/29/20				1.1.1, 1.1.2	Open
1.2.1	Rejects-handling engine design	Megan	9	Fri 1/17/20	Wed 1/29/20					Open
1.3	Develop OLAP design document	Joachim	9	Fri 1/17/20	Wed 1/29/20				1.1.1, 1.1.2	Open
1.3.1	Universe design	Joachim	9	Fri 1/17/20	Wed 1/29/20					Open
1.4	Develop OLAP design part 1	Kevin	8	Fri 1/10/20	Tues 1/21/20					Open
1.4.1	High-priority reports design	Kevin	8	Fri 1/10/20	Tues 1/21/20					Open
1.5	Develop application design document	Tomas	9	Fri 1/17/20	Wed 1/29/20					Open
1.5.1	Group consolidation and corporate reporting (GCCR) maintenance application design	Tomas	9	Fri 1/17/20	Wed 1/29/20					Open
1.6	Extract, transform, load (ETL) design document	Joachim	2	Thu 1/30/20	Fri 1/31/20				1.5	Open
1.6.1	Data export utility design	Joachim	2	Thu 1/30/20	Fri 1/31/20					Open
1.7	Application design document	Mei-ling	27	Mon 1/6/20	Tue 2/11/20					Open
1.7.1	Web entry application UI design	Mei-ling	26	Mon 1/6/20	Wed 2/10/20					Open
1.7.2	Web entry application UI design sign-off	Mei-ling	1	Tue 2/11/20	Tue 2/11/20					Open
1.7.3	Web entry forms and database model validation	Kevin	11	Wed 1/15/20	Wed 1/29/20					Open
1.8	Functional requirements document	Chantelle	9	Mon 1/20/20	Thu 1/30/20					Open
1.8.1	Application design	Chantelle	9	Mon 1/20/20	Thu 1/30/20					Open
1.8.1.1	User authentication	Chantelle	4	Mon 1/20/20	Thu 1/23/20					Open
1.8.1.2	Call logging	Chantelle	2	Fri 1/24/20	Mon 1/27/20					Open
1.8.1.3	Search	Chantelle	3	Tue 1/28/20	Thu 1/30/20					Open

(Thanks to Priya Padmanhabhan for suggesting this example.)

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

Motivation (1 of 2)

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

Motivation (2 of 2)

- Consider the “de-motivators” … DO NOT
  - Assign unrealistic deadlines
  - Ignore good efforts
  - Accept a low-quality product
  - Give everyone on the project the same raise
  - Make an important decision without the team’s input
  - Maintain poor working conditions

Assuring Group Performance

- Make sure team understands the project and its goals
- Establish operating procedures (Project Charter)
  - Availability
  - Status reporting
  - Meetings
- Ensure that team members get to know each other
- Establish methods for dealing with problems

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

Source: Barry W. Boehm and colleagues, "Cost Models for Future Software Life Cycle Processes: COCOMO 2.0," in J. D. Arthur and S. M. Henry (eds.) *Annals of Software Engineering Special Volume on Software Process and Product Measurement*, Amsterdam: J. C. Baltzer AG Science Publishers, 1995.

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

Managing Scope

- Beware of scope creep
- Use JAD and prototyping to minimize scope creep pressure
- Implement formal change approval process
- Defer additional requirements as future system enhancements

Timeboxing

- 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

When a Target Date is Missed…

- Don’t assume you can catch up
- The ONLY situation in which you can make up time is when:
  - The remainder of the project is simpler than the part you fell behind on, and
  - The remainder of the project is simpler than you expected when the original estimates were made.
- Evaluate the complexity of the remainder of the project to determine the correct schedule adjustment.
- Adding people is not always the right way to handle schedule slippages.

After reading and studying this chapter, you should be able to: (1 of 3)

- 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.

After reading and studying this chapter, you should be able to: (2 of 3)

- List and explain two Agile methodologies.
- For each methodology included in the chapter, be able to summarize the project characteristics that make that methodology the best choice and the poorest choice. Be able to explain why.
- Discuss ways to estimate the time frame for a project.
- Discuss the purpose and content of the project work plan.
- Discuss the three main tasks involved when staffing a project.
- Describe various ways to influence the motivation of project team members.

After reading and studying this chapter, you should be able to: (3 of 3)

- 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.
- 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.

Copyright