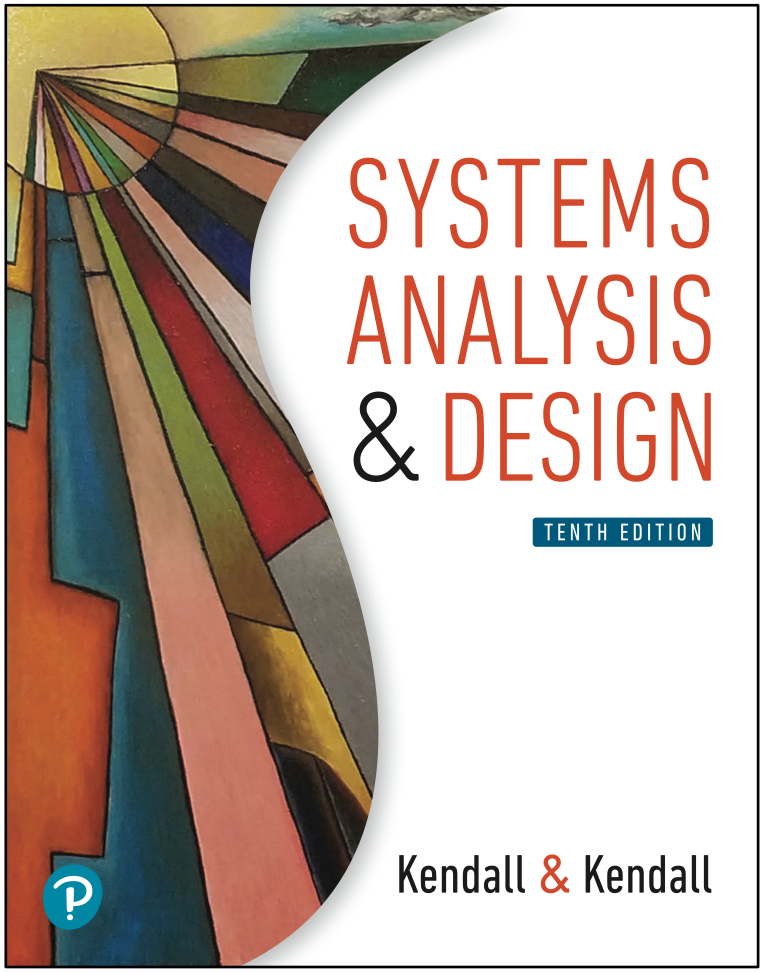
Systems Analysis & Design

Tenth Edition



Chapter 6

Agile Modeling,
Prototyping, and Scrum

Learning Objectives (1 of 2)

6.1 Understand the roots of agile modeling in prototyping and the four main types of prototyping

6.2 Understand agile modeling and the core practices that differentiate it from other development methodologies

6.3 Understand Scrum as an agile method to improve development of complex projects

Learning Objectives (2 of 2)

6.4 Learn about DevOps as a cultural shift in the way to organize rapid systems development and operations

6.5 Learn the importance of values critical to agile modeling

6.6 Understand how to improve efficiency for users who are knowledge workers using either structured methods or agile modeling

Agile Modeling, but First Prototyping

- Agile modeling is a collection of innovative, user-centered approaches to system development
- Prototyping is an information-gathering technique useful in seeking
  - User reactions
  - Suggestions
  - Innovations
  - Revision plans

Major Topics

- Prototyping
- Agile modeling
- Scrum
- DevOps (derived from Development and Operations)

Prototyping

- Patched-up
- Nonoperational
- First-of-a-series
- Selected features

Patched-Up Prototype

- A system that works but is patched up or patched together
- A working model that has all the features but is inefficient
- Users can interact with the system
- Retrieval and storage of information may be inefficient

Nonoperational Scale Models

- A nonworking scale mode that is set up to test certain aspects of the design
- A nonworking scale model of an information system might be produced when the coding required by the application is too expensive to prototype but when a useful idea of the system can be gained through prototyping of the input and output only

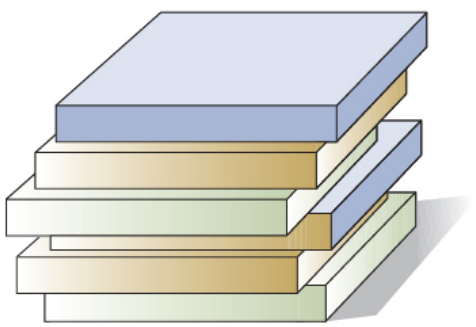
First-Of-A-Series Prototype

- Creating a pilot
- Prototype is completely operational
- Useful when many installations of the same information system are planned
- A full-scale prototype is installed in one or two locations first, and if successful, duplicates are installed at all locations based on customer usage patterns and other key factors

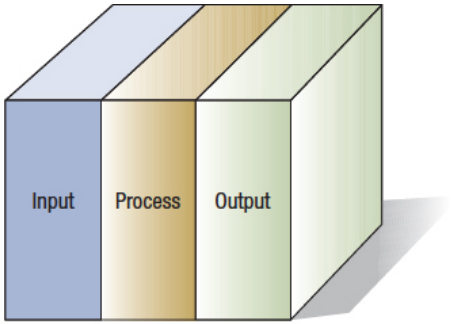
Selected Features Prototype

- Building an operational model that includes some, but not all, of the features that the final system will have
- Some, but not all, essential features are included
- Built in modules
- Part of the actual system

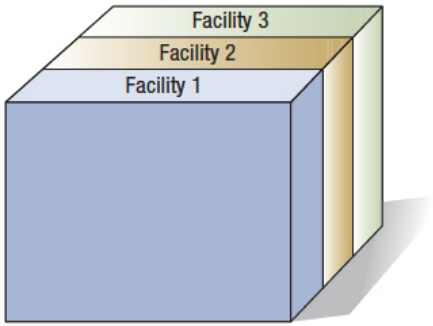
Figure 6.1 Four Kinds of Prototypes



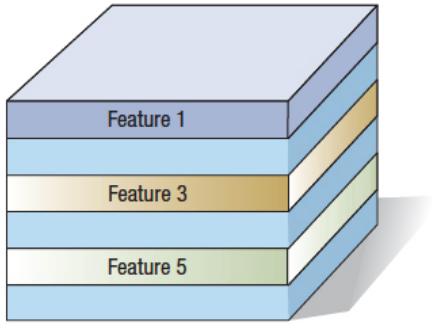
Patched-Up Prototype



Nonoperational Prototype



First-of-a-Series Prototype



Selected Features Prototype

Users' Role in Prototyping

- Honest involvement
- Communicate the purpose of the prototype clearly
- Get suggestions for changing, expanding the prototype and innovation

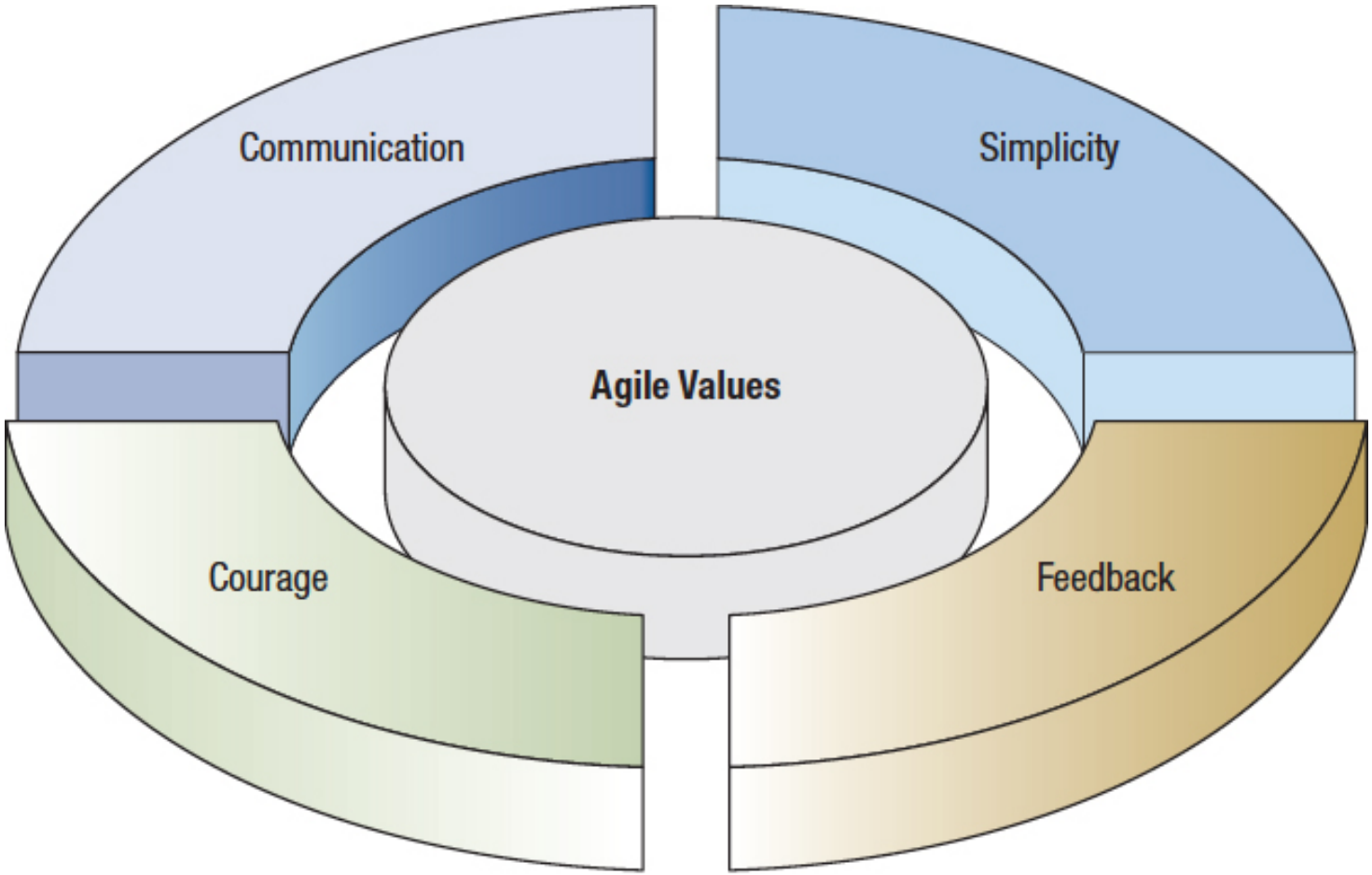
Agile Modeling

- Agile methods are a collection of innovative, user-centered approaches to systems development

Values and Principles of Agile Modeling

- Communication
- Simplicity
- Feedback
- Courage

Figure 6.2 Values are Crucial to the Agile Approach



The Basic Principles of Agile Modeling (1 of 3)

- Satisfy the customer through delivery of working software
- Embrace change, even if introduced late in development
- Continue to deliver functioning software incrementally and frequently
- Encourage customers and analysts to work together daily
- Trust motivated individuals to get the job done

The Basic Principles of Agile Modeling (2 of 3)

- Promote face-to-face conversation
- Concentrate on getting software to work
- Encourage continuous, regular, and sustainable development
- Adopt agility with attention to mindful design
- Support self-organizing teams

The Basic Principles of Agile Modeling (3 of 3)

- Provide rapid feedback
- Encourage quality
- Review and adjust behavior occasionally
- Adopt simplicity

Four Basic Activities of Agile Modeling

- Coding
- Testing
- Listening
- Designing

Coding

- Coding is the one activity that it is not possible to do without
- The most valuable thing that we receive from code is “learning”
- Code can also be used to communicate ideas that would otherwise remain fuzzy or unshaped

Testing (1 of 2)

- Automated testing is critical
- Write tests to check coding, functionality, performance, and conformance
- Use automated tests
- Large libraries of tests exist for most programming languages
- These are updated as necessary during the project

Testing (2 of 2)

- Testing in the short term gives extreme confidence in what you are building
- Testing in the long term keeps a system alive and allows for changes longer than would be possible if no tests were written or run

Listening (1 of 2)

- Listening is done in the extreme
- Developers use active listening to hear their programming partner
- Because there is less reliance on formal, written communication, listening becomes a paramount skill

Listening (2 of 2)

- A developer also uses active listening with the customer
- Developers assume that they know nothing about the business so they must listen carefully to businesspeople

Designing (1 of 2)

- Designing is a way of creating a structure to organize all the logic in the system
- Designing is evolutionary, and so systems are conceptualized as evolving, always being designed
- Good design is often simple

Designing (2 of 2)

- Design should allow flexibility
- Effective design locates logic near the data on which it will be operating
- Design should be useful to all those who will need it as the development effort proceeds

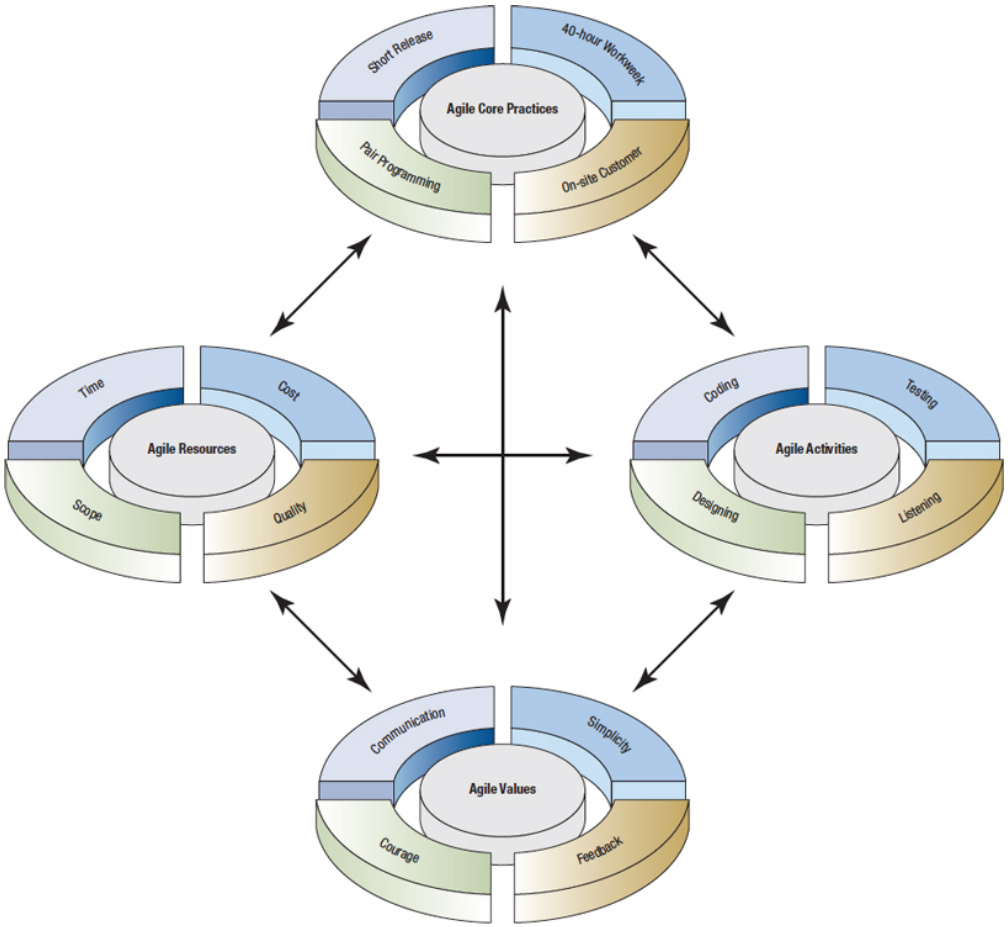
Four Resource Control Variables of Agile Modeling

- Time
- Cost
- Quality
- Scope

Four Core Agile Practices

- Short releases
- 40-hour work week
- Onsite customer
- Pair programming

Figure 6.3 Agile Core Practices



The Agile Development Process

- Listen for user stories
- Draw a logical workflow model
- Create new user stories based on the logical model
- Develop some display prototypes
- Create a physical data model using feedback from the prototypes and logical workflow diagrams

Writing User Stories

- Spoken interaction between developers and users
- Seeking first and foremost to identify valuable business user requirements
- The goal is prevention of misunderstandings or misinterpretations of user requirements

Figure 6.4 User Stories Can Be Recorded on Cards

Need or Opportunity:	Apply shortcut methods for faster checkout.					
Story:	If the identity of the customer is known and the delivery address matches, speed up the transaction by accepting the credit card on file and the rest of the customer's preferences such as shipping method.					
		Well Below	**Below Average**	**Average**	**Above Average**	**Well Above**
Activities:	**Coding**			✓		
	Testing			✓		
	Listening			✓		
	Designing				✓	
Resources:	**Time**				✓	
	Cost		✓			
	Quality				✓	
	Scope			✓		

Scrum (1 of 2)

- Begin the project with a high-level plan that can be changed on the fly
- Success of the project is most important
- Individual success is secondary
- Project leader has some (not much) influence on the detail
- Systems team works within a strict time frame (two to four weeks for development)

Roles Played in Scrum

- There are three roles in Scrum:
  - Product owner
  - Scrum Master
  - Team member

Team Members

- Team members:
  - Work to create and improve user stories
  - Generate estimates
  - Self-organize to complete the work
  - Exhibit willingness to participate in any activity to help the project

Scrum (2 of 2)

- Product backlog
- Sprint backlog
- Sprint
- Daily scrum
- Demo

Product Backlog

- Features and other deliverables designers intend for the product based on user stories
- The list of user stories is reorganized so that the most important user stories appear on the top

Figure 6.5 A Product Backlog Registry

Number	Tasks based on user stories	Who will benefit	Resources required	Will be accepted when
W212	Make it easier to continue shopping once an item is placed in the shopping cart	Visitors to the website	Minor coding effort, testing	Users react positively to the prototype, coding is completed and tested
W210	Offer options for shipping earlier in the online ordering process	Visitors to the website	Dynamically calculating rates based on zip code and preferred delivery speed, coding, testing	Validity is checked, code is tested, information is approved as understandable to users
W211	Add product reviews by peers	Users who would feel more confident having these reviews available	Collaborative filtering, redesign of product pages, coding, testing	Users approve of the prototype and the collaborative system is thoroughly tested

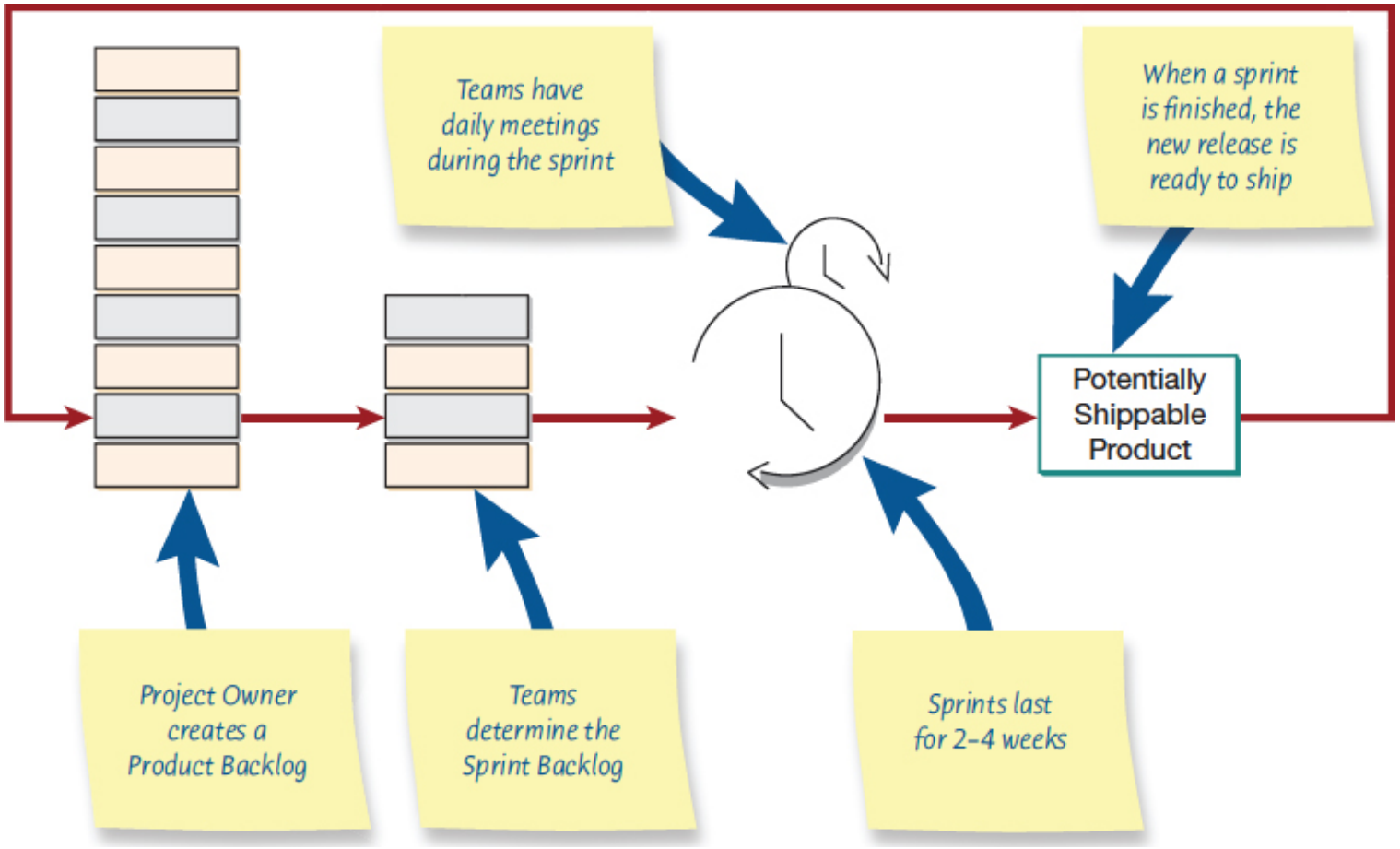
Sprint Cycle

- Stories are deliverables the team accomplishes
- Tasks are parts of the story or units of work that each team members does
- The sprint cycle can vary in length, but the usual is two weeks
- At the end of the cycle, determine whether the product should be released

Two-Week Release Advantages

- Team spirit remains high
- Completing the product is more real to the team
- Continual feedback from customers

Figure 6.6 Teams Work Together to Accomplish a Sprint Cycle



Lessons Learned from Agile Modeling (1 of 2)

- Short releases allow the system to evolve
- Pair programming enhances the overall quality
- Onsite customers are mutually beneficial to the business and the agile development team

Other Unique Scrum Features

- Scrum planning meeting
- Planning poker
- Daily meetings
- Sprint burndown chart
- Sprint review

Scrum Planning Meeting

- Two parts to a Scrum planning meeting:
  - Product owner presents the list of features on the wish list of user stories
  - Estimate the resources needed to complete all of the features
    - Common way to do this is to play planning poker

Planning Poker

- Planning poker is a way to help the team determine estimates for completing the features from user stories
- There are rules for the game

Planning Poker Rules (1 of 3)

- A moderator keeps the estimation meeting running efficiently
- The product owner presents one of the user stories that needs estimating
- Each team member chooses a card with a number on it and lays it face down
- A lower numbered card means that a project can be completed faster

Planning Poker Rules (2 of 3)

- The numbers may represent the number of days it takes to finish a feature, story points, or even difficulty in abstract terms
- Team members turn over their cards at the same time

Planning Poker Rules (3 of 3)

- The team members with the highest and lowest estimates may defend their ratings
- The entire team can discuss the estimates
- A new hand is then played and so on

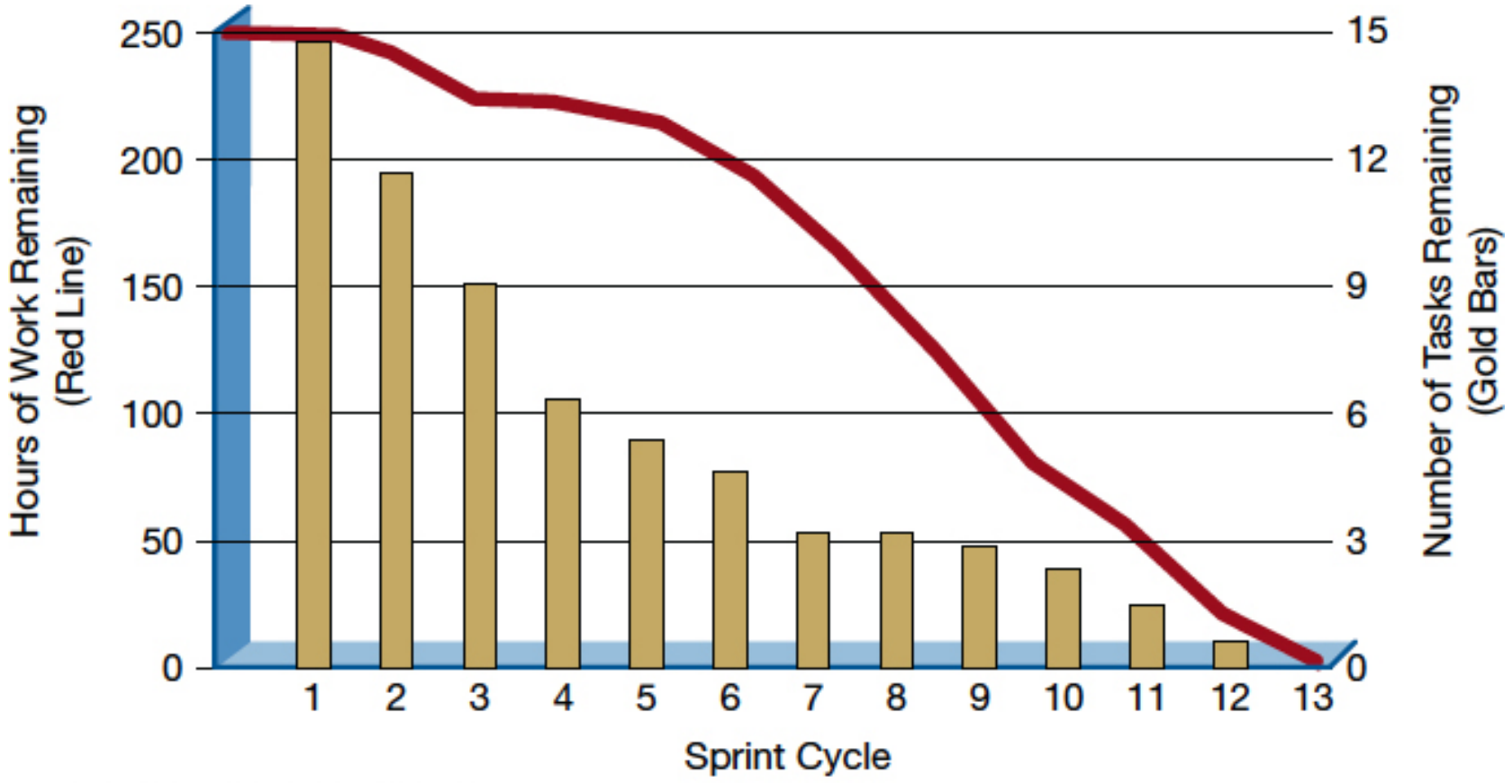
Daily Scrum Meetings

- Called a stand-up meeting because it lasts only a few minutes
- Team members tell each other what tasks they've been working on since the previous daily Scrum
- Tasks they expect to complete during the current daily Scrum
- Obstacles might get in their way

Sprint Burndown Chart

- Way to keep track of performance
  - Horizontal axis tracks the time that has elapsed
  - Vertical axis may track the number of tasks remaining or the number of hours to complete the remaining tasks
  - Red line shows hours of work remaining, and the yellow bars show the number of tasks remaining

Figure 6.8 A Burndown Chart



Lessons Learned from Agile Modeling (2 of 2)

- The 40-hour work week improves worker effectiveness
- Balanced resources and activities support project goals
- Agile values are crucial to success

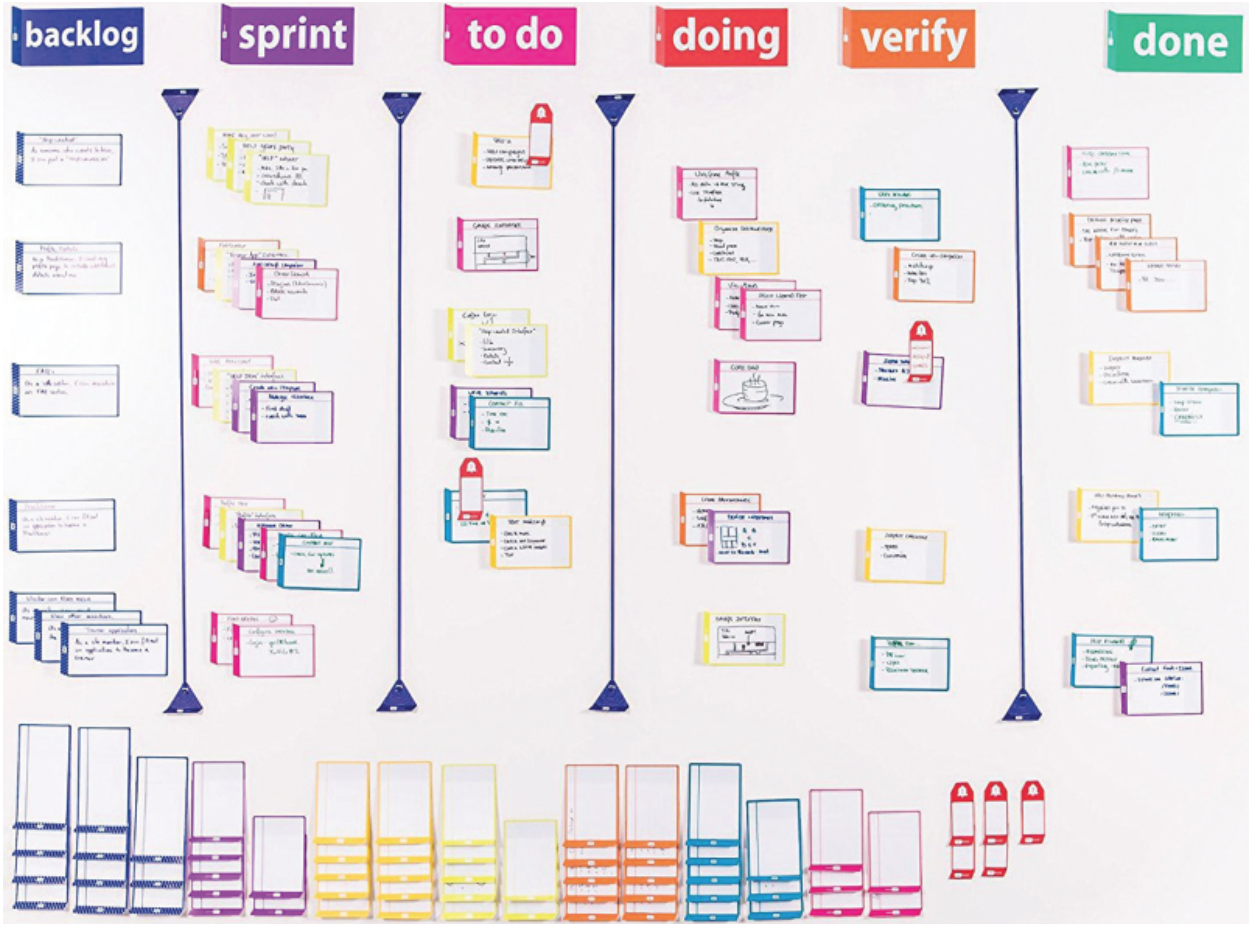
Sprint Review

- Team gets together in a meeting to review the work done
- Note any tasks that were not completed
- User stories completed are prominently documented
- User stories the team committed to that were not finished are noted

Kanban

- Key elements of the Kanban system as applied to software development are:
  - Visualize the workflow
  - Keep work-in-process (WIP) as small as possible
  - Reevaluate the workflow, reassigning priorities if need be
  - Strive for continual improvement

Figure 6.9 A Kanban Board



Scrum Advantages (1 of 2)

- Advantages:
  - Quick product development
  - Exercises a user-oriented approach
  - Encourages teamwork
  - Less confusing than more formal approaches
  - Flexibility

Scrum Advantages (2 of 2)

- Advantages:
  - Satisfying to team members
  - Rewards smaller but meaningful accomplishments
  - Provides feedback
  - Adaptability

Scrum Disadvantages

- Disadvantages:
  - Documenting features improperly
  - Releasing products with errors
  - Releasing products too soon for the user
  - Completing the sprint backlog under pressure
  - Working as a geographically dispersed team may be difficult
  - Working as a team when special skills are required may be challenging
  - Replacing team members who leave the team is difficult

Devops – Development Operations

- Decrease the deployment time for newly developed applications and maximize profit by rapidly addressing market opportunities and getting customer feedback in a timely manner
- Development and operations are working in parallel streams within a DevOps culture

Figure 6.10 Devops Exists in Its Own Culture

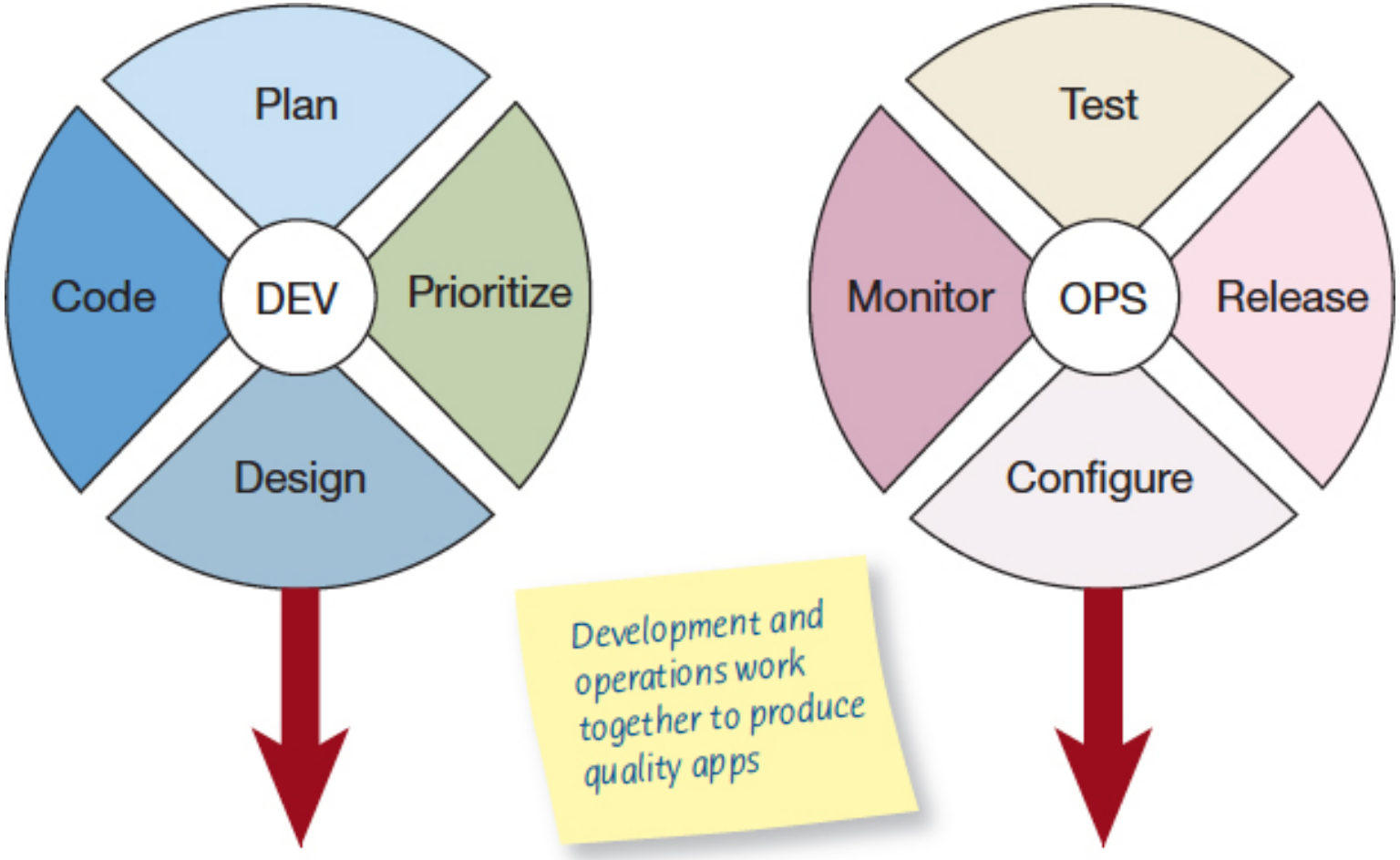
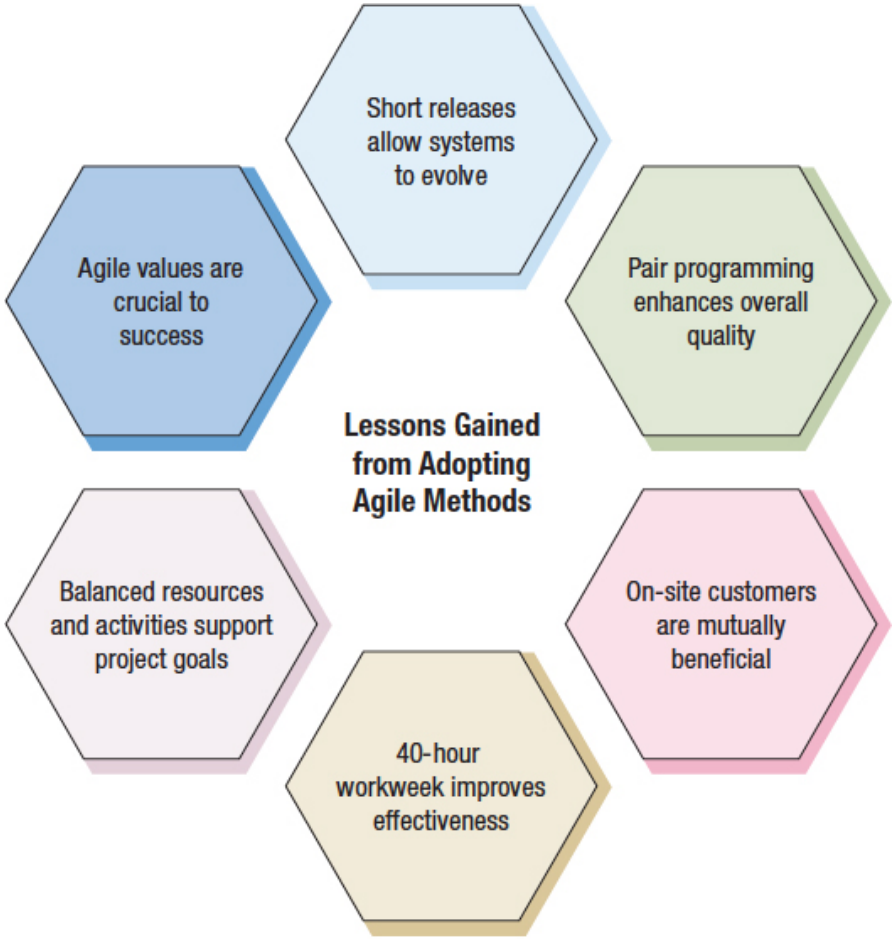


Figure 6.11 There are Six Vital Lessons That Can Be Drawn from the Agile Approach to Systems



Comparing Agile Modeling and Structured Methods

- Improving the efficiency of systems development
- Risks inherent in organizational innovation

Figure 6.12 Strategies for Improving Efficiency can be Implemented Using Two Different Development Approaches

Strategies for Improving Efficiency in Knowledge Work	Implementation Using Structured Methodologies	Implementation Using Agile Methodologies
Reduce interface time and errors	Adopting organization standards for coding, naming, etc.; using forms	Adopting pair programming
Reduce process learning time and dual processing losses	Managing when updates are released so the user does not have to learn and use software at the same time	Ad hoc prototyping and rapid development
Reduce time and effort to structure tasks and format outputs	Using CASE tools and diagrams; using code written by other programmers	Encouraging short releases
Reduce nonproductive expansion of work	Managing project; establishing deadlines	Limiting scope in each release
Reduce data and knowledge search and storage time and costs	Using structured data gathering techniques, such as interviews, observation, sampling	Allowing for an on-site customer
Reduce communication and coordination time and costs	Separating projects into smaller tasks; establishing barriers	Timeboxing
Reduce losses from human information overload	Applying filtering techniques to shield analysts and programmers	Sticking to a 40-hour workweek

Figure 6.13 Adopting New Information Systems Involves Balancing Several Risks

Risks When Adopting New Information Systems

- Fit of development team culture
- Best time to innovate
- Training cost for analysts and programmers
- Client's reaction to new methodology
- Impact of agile methodologies
- Programmers/analysts individual rights

Summary (1 of 3)

- Prototyping
  - Patched-up system
  - Nonoperational
  - First-of-a-series
  - Selected-features

Summary (2 of 3)

- Agile modeling
- Five values of the agile approach
- Principles of agile development
- Agile activities
- Agile resources
- Core practices of the agile approach
- Stages in the agile development process

Summary (3 of 3)

- User stories
- Agile lessons
- Scrum methodology
- DevOps
- Dangers to adopting innovative approaches

Copyright