1

Kendall & Kendall
Systems Analysis and Design, 9e

Systems, Roles, and Development Methodologies

Learning Objectives

- Understand the need for systems analysis and design in organizations.
- Realize what the many roles of the systems analyst are.
- Comprehend the fundamentals of three development methodologies:
  - SDLC
  - The agile approach
  - Object-oriented systems analysis and design

Information—A Key Resource

- Fuels business and can be the critical factor in determining the success or failure of a business
- Needs to be managed correctly
- Managing computer-generated information differs from handling manually produced data

Major Topics

- Fundamentals of different kinds of information systems
- Roles of systems analysts
- Phases in the systems development life cycle as they relate to Human-Computer Interaction (HCI) factors
- CASE tools
- Open Source Software

Need for Systems Analysis and Design

- Installing a system without proper planning leads to great user dissatisfaction and frequently causes the system to fall into disuse
- Lends structure to the analysis and design of information systems
- A series of processes systematically undertaken to improve a business through the use of computerized information systems

Roles of the Systems Analyst

- The analyst must be able to work with people of all descriptions and be experienced in working with computers
- Three primary roles:
  - Consultant
  - Supporting expert
  - Agent of change

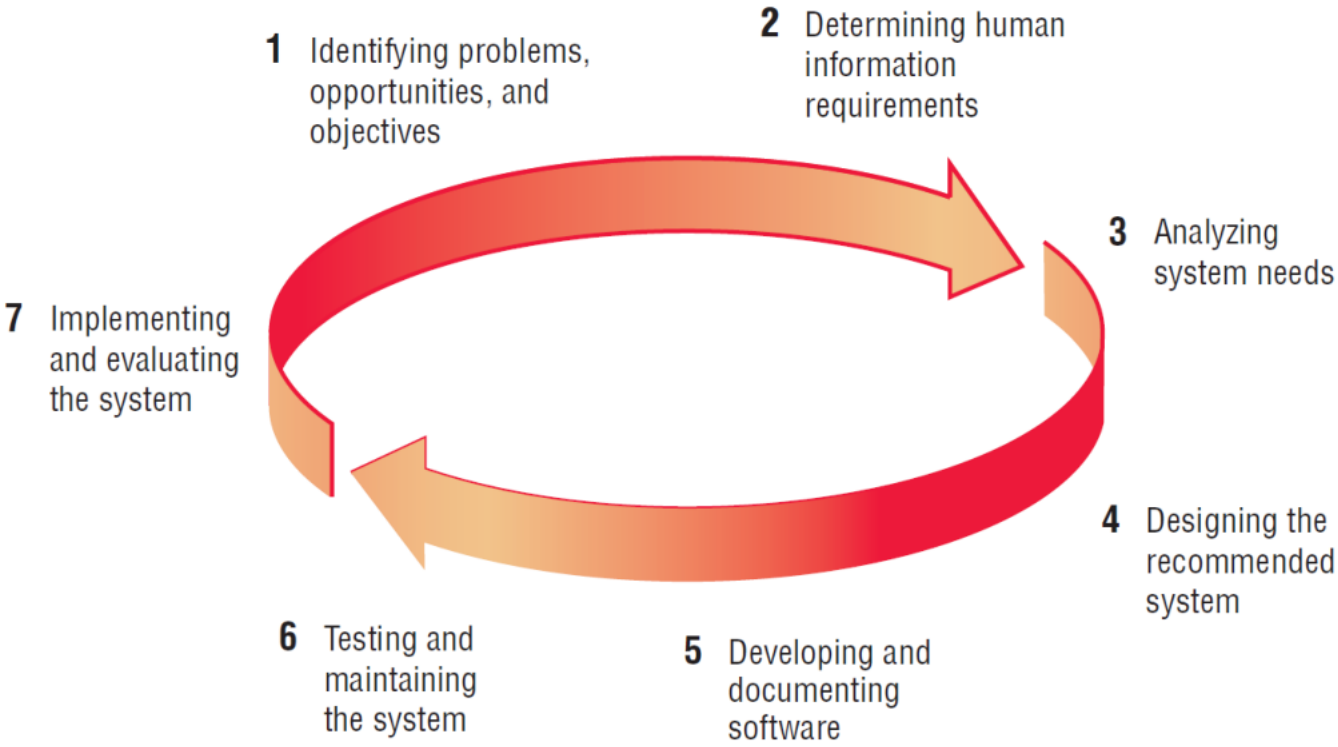
Qualities of the Systems Analyst

- Problem solver
- Communicator
- Strong personal and professional ethics
- Self-disciplined and self-motivated

Systems Development Life Cycle (SDLC)

- The systems development life cycle is a phased approach to solving business problems
- Developed through the use of a specific cycle of analyst and user activities
- Each phase has unique user activities

The Seven Phases of the Systems Development Life Cycle (Figure 1.1)



1 Identifying problems, opportunities, and objectives

2 Determining human information requirements

3 Analyzing system needs

4 Designing the recommended system

5 Developing and documenting software

6 Testing and maintaining the system

7 Implementing and evaluating the system

Incorporating Human-Computer Interaction (HCI) Considerations

- The demand for analysts who are capable of incorporating HCI into the systems development process keeps increasing, as companies begin to realize that the quality of systems and the quality of work life can be improved by taking a human-centered approach at the outset of a project

Identifying Problems, Opportunities, and Objectives

- Activity:
  - Interviewing user management
  - Summarizing the knowledge obtained
  - Estimating the scope of the project
  - Documenting the results

Identifying Problems, Opportunities, and Objectives

- Output:
  - Feasibility report containing problem definition and objective summaries from which management can make a decision on whether to proceed with the proposed project

Determining Human Information Requirements

- Activity:
  - Interviewing
  - Sampling and investing hard data
  - Questionnaires
  - Observe the decision maker’s behavior and environment
  - Prototyping
  - Learn the who, what, where, when, how, and why of the current system

Determining Human Information Requirements

- Output:
  - The analyst understands how users accomplish their work when interacting with a computer
  - Begin to know how to make the new system more useful and usable
  - Know the business functions
  - Have complete information on the:
    - People
    - Goals
    - Data
    - Procedure involved

Analyzing System Needs

- Activity:
  - Create data flow, activity, or sequence diagrams
  - Complete the data dictionary
  - Analyze the structured decisions made
  - Prepare and present the system proposal
- Output:
  - Recommendation on what, if anything, should be done

Designing the Recommended System

- Activity:
  - Design procedures for data entry
  - Design the human-computer interface
  - Design system controls
  - Design database and/or files
  - Design backup procedures
- Output
  - Model of the actual system

Developing and Documenting Software

- Activity:
  - System analyst works with programmers to develop any original software
  - Works with users to develop effective documentation
  - Programmers design, code, and remove syntactical errors from computer programs
  - Document software with help files, procedure manuals, and Web sites with Frequently Asked Questions

Developing and Documenting Software

- Output:
  - Computer programs
  - System documentation

Testing and Maintaining the System

- Activity:
  - Test the information system
  - System maintenance
  - Maintenance documentation
- Output:
  - Problems, if any
  - Updated programs
  - Documentation

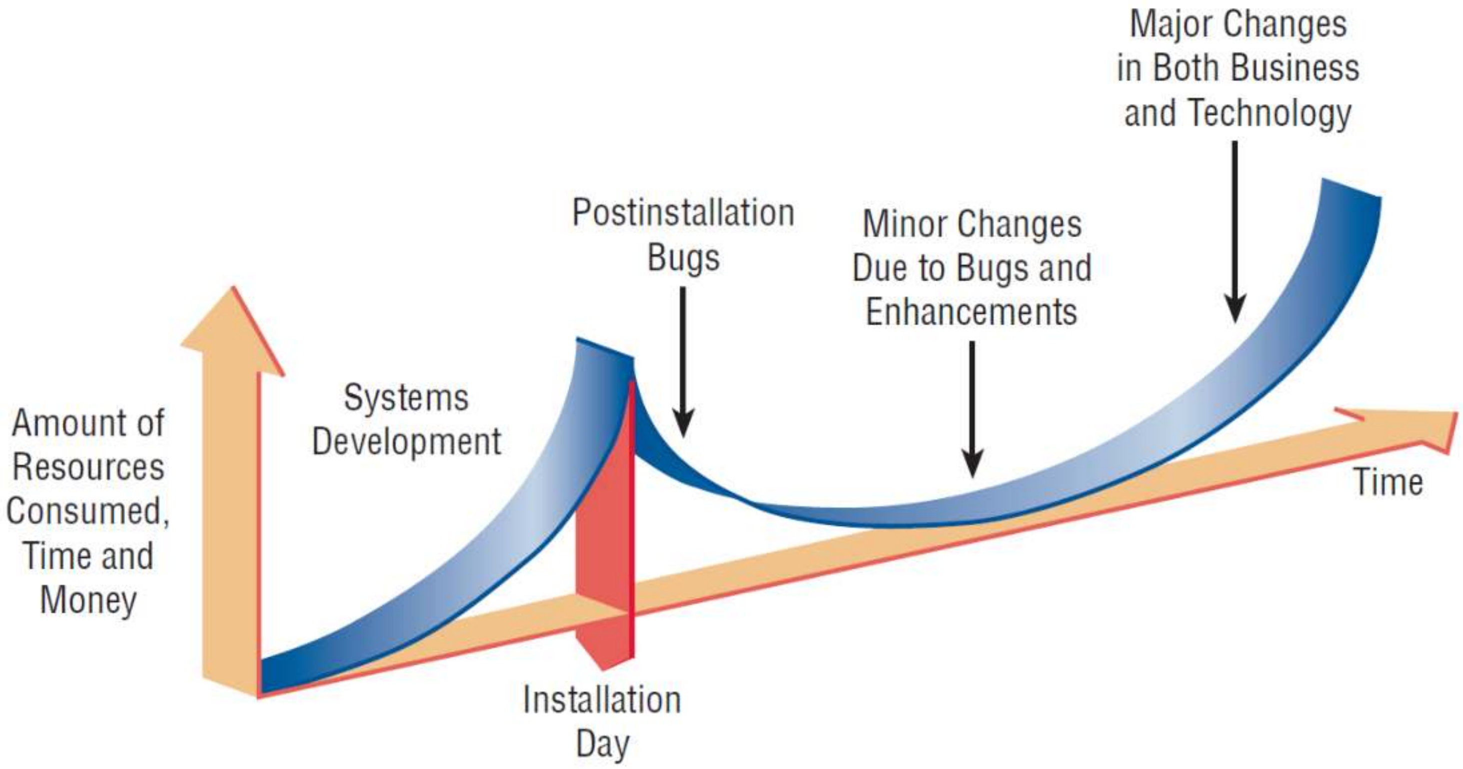
Implementing and Evaluating the System

- Activity:
  - Train users
  - Analyst plans smooth conversion from old system to new system
  - Review and evaluate system
- Output:
  - Trained personnel
  - Installed system

The Impact of Maintenance

- Maintenance is performed for two reasons:
  - Removing software errors
  - Enhancing existing software
- Over time the cost of continued maintenance will be greater than that of creating an entirely new system. At that point it becomes more feasible to perform a new systems study.

Resource Consumption over the System Life (Figure 1.3)


Major Changes in Both Business and Technology
Postinstallation Bugs
Minor Changes Due to Bugs and Enhancements
Systems Development
Amount of Resources Consumed, Time and Money
Time
Installation Day

Approaches to Structured Analysis and Design and to the Systems Development Life Cycle

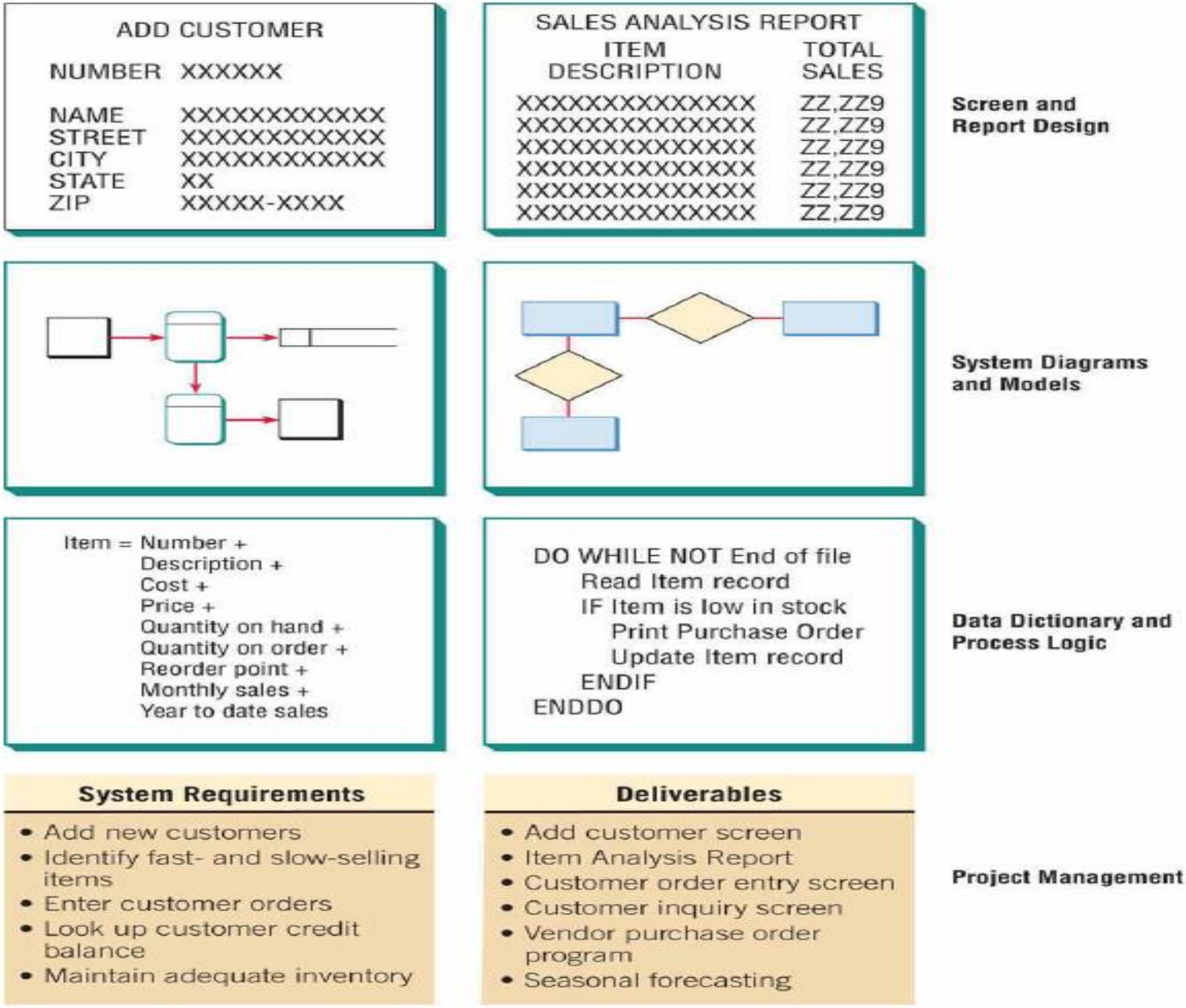
- Traditional systems development life cycle
- CASE systems development life cycle
- Object-oriented systems analysis and design

Case Tools

- CASE tools are productivity tools for systems analysts that have been created explicitly to improve their routine work through the use of automated support

Reasons for Using Case Tools

- Reasons for using CASE tools
  - Increasing analyst productivity
  - Improving analyst-user communication
  - Integrating life cycle activities



```
ADD CUSTOMER

NUMBER  XXXXXX

NAME    XXXXXXXXXXXX
STREET  XXXXXXXXXXXX
CITY    XXXXXXXXXXXX
STATE   XX
ZIP     XXXXX-XXXX
```

```
SALES ANALYSIS REPORT
      ITEM              TOTAL
   DESCRIPTION          SALES
XXXXXXXXXXXXXX          ZZ,ZZ9
XXXXXXXXXXXXXX          ZZ,ZZ9
XXXXXXXXXXXXXX          ZZ,ZZ9
XXXXXXXXXXXXXX          ZZ,ZZ9
XXXXXXXXXXXXXX          ZZ,ZZ9
XXXXXXXXXXXXXX          ZZ,ZZ9
```

Screen and Report Design

System Diagrams and Models

```
Item = Number +
       Description +
       Cost +
       Price +
       Quantity on hand +
       Quantity on order +
       Reorder point +
       Monthly sales +
       Year to date sales
```

```
DO WHILE NOT End of file
    Read Item record
    IF Item is low in stock
        Print Purchase Order
        Update Item record
    ENDIF
ENDDO
```

Data Dictionary and Process Logic

System Requirements

- Add new customers
- Identify fast- and slow-selling items
- Enter customer orders
- Look up customer credit balance
- Maintain adequate inventory

Deliverables

- Add customer screen
- Item Analysis Report
- Customer order entry screen
- Customer inquiry screen
- Vendor purchase order program
- Seasonal forecasting

Project Management

The Agile Approach

- Based on:
  - Values
  - Principles
  - Core practices

Agile Values

- Communication
- Simplicity
- Feedback
- Courage

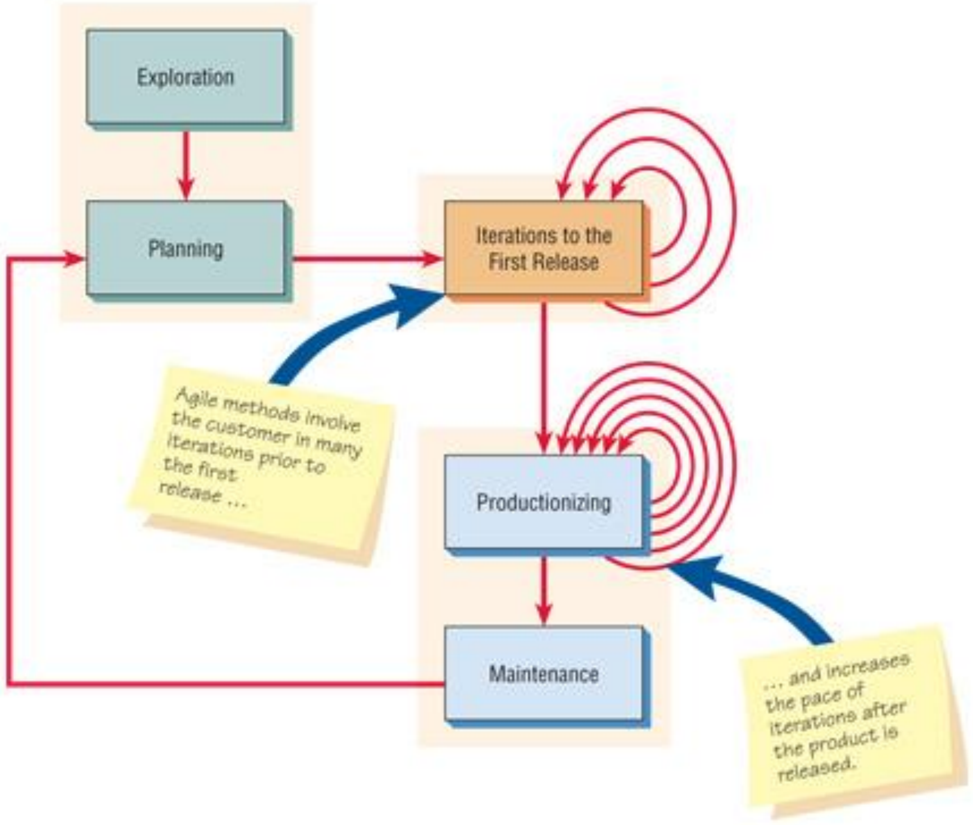
Four Agile Resources

- Resources are adjusted to ensure successful project completion
  - Time
  - Cost
  - Quality
  - Scope

Five Stages of Agile Development

- Exploration
- Planning
- Iterations to the first release
- Productionizing
- Maintenance

Agile Project Development Process (Figure 1.5)


Exploration
Planning
Iterations to the First Release
Agile methods involve the customer in many iterations prior to the first release ...
Productionizing
Maintenance
... and increases the pace of iterations after the product is released.

Object-Oriented (O-O) Systems Analysis and Design

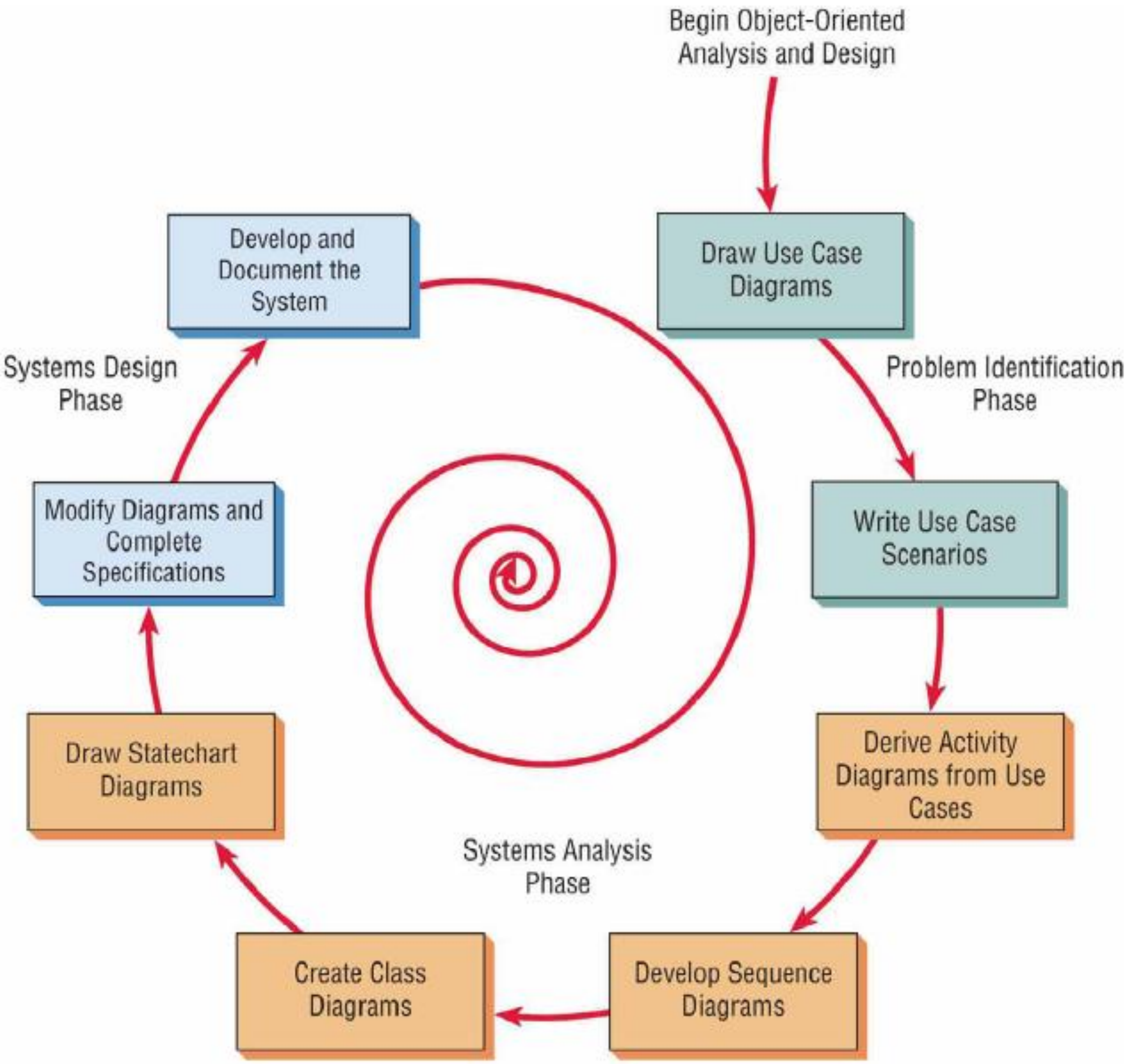
- Alternate approach to the structured approach of the SDLC that is intended to facilitate the development of systems that change rapidly in response to dynamic business environments
- Analysis is performed on a small part of the system followed by design and implementation

Object-Oriented (O-O) Systems Analysis and Design

- The cycle repeats with analysis, design, and implementation of the next part and this repeats until the project is complete
- Examines the objects of a system

Unified Modeling Language (UML) Phases

- Define the use case model:
  - Use case diagram
  - Use case scenarios
- Create UML diagrams
- Develop class diagrams
- Draw statechart diagrams
- Modify the UML diagrams
- Develop and document the system


Begin Object-Oriented Analysis and Design
Draw Use Case Diagrams
Problem Identification Phase
Write Use Case Scenarios
Derive Activity Diagrams from Use Cases
Develop Sequence Diagrams
Systems Analysis Phase
Create Class Diagrams
Draw Statechart Diagrams
Modify Diagrams and Complete Specifications
Systems Design Phase
Develop and Document the System

Choosing a Method

- Choose either:
    - SDLC
    - Agile
    - Object-oriented methodologies

When to Use SDLC

- Systems have been developed and documented using SLDC
- It is important to document each step
- Upper level management feels more comfortable or safe using SDLC
- There are adequate resources and time to complete the full SDLC
- Communication of how new systems work is important

When to Use Agile

- There is a project champion of agile methods in the organization
- Applications need to be developed quickly in response to a dynamic environment
- A rescue takes place (the system failed and there is no time to figure out what went wrong)
- The customer is satisfied with incremental improvements
- Executives and analysts agree with the principles of agile methodologies

When to Use Object-Oriented

- The problems modeled lend themselves to classes
- An organization supports the UML learning
- Systems can be added gradually, one subsystem at a time
- Reuse of previously written software is a possibility
- It is acceptable to tackle the difficult problems first

Open Source Software

- An alternative of traditional software development where proprietary code is hidden from the users
- Open source software is free to distribute, share, and modify
- Characterized as a philosophy rather than simply the process of creating new software
- Examples: Linux Operating System, Apache Web Server, Mozilla Firefox

Four Types of Open Source Communities:

- Ad hoc
- Standardized
- Organized
- Commercial

Six Key Dimensions that Differentiate Open Source Communities

- General structure
- Environment
- Goals
- Methods
- User community
- Licensing

Reasons for Participating in Open Source Communities

- Rapidity with which new software can be developed and tested
- Faster to have a committed group of experts develop, test, and debug code
- This fosters creativity
- Have many good minds work with innovative applications

Reasons for Participating in Open Source Communities

- Potential to reduce development costs
- Bolster their self-image
- Contribute something worthwhile to the software development community

Open Source Contribution and Differentiation

- Contributions to the open community and differentiation from the open community are for the following reasons:
  - Cost
  - Managing resources
  - Time it takes to bring a new product to the market

Reasons for Analyst Participation in the Open Source Community

- Curiosity about software benefits
- Achieve collective design
  - Incorporate open source software design into:
    - Proprietary products
    - Processes
    - Knowledge
    - IT artifacts

Collective Design

- Through a process of collective design the IT artifact is imbued with
  - Community and organizational structures
  - Knowledge
  - Practices

Summary

- Information is a key resource
- Integration of traditional systems with new technologies
- Roles and qualities of the systems analyst
- The systems development life cycle
- CASE tools
- Agile systems development
- Object-oriented systems development
- Open source systems