2

PENDEKATAN DALAM PEMBANGUNAN SISTEM

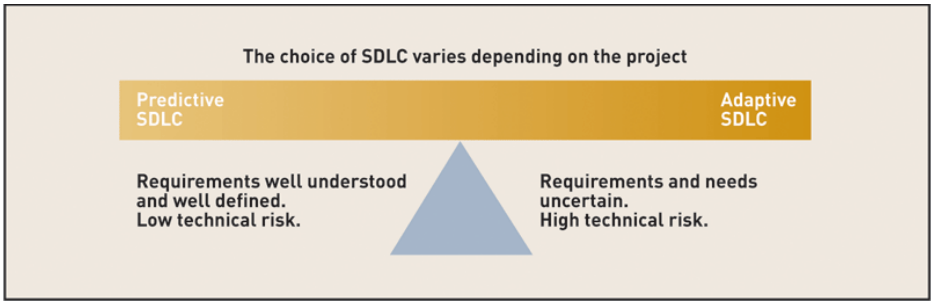
2

The Systems Development Lifecycle (SDLC)

- Systems development life cycle (SDLC)
  - Provides overall framework for managing systems development process
- Two main approaches to SDLC
  - Predictive approach – assumes project can be planned out in advance
  - Adaptive approach – more flexible, assumes project cannot be planned out in advance
- All projects use some variation of SDLC

2

Choosing the Predictive vs. Adaptive Approach to the SDLC (Figure 2-1)



3

2

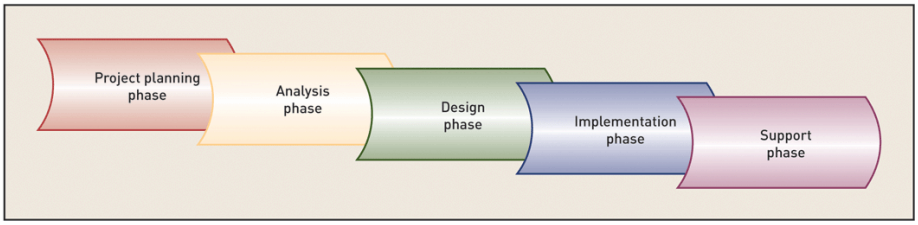
Traditional Predictive Approach to the SDLC

- Project planning – initiate, ensure feasibility, plan schedule, obtain approval for project
- Analysis – understand business needs and processing requirements
- Design – define solution system based on requirements and analysis decisions
- Implementation – construct, test, train users, and install new system
- Support – keep system running and improve

4

Information System Development Phases

Figure 2-2
Information system development phases



SDLC and Problem Solving

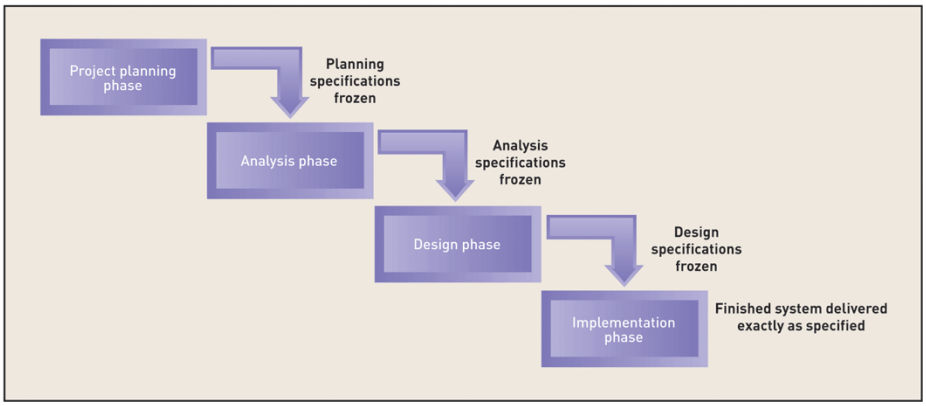
- Similar to problem-solving approach in Chapter 1
  - Organization recognizes problem (project planning)
  - Project team investigates, understands problem and solution requirements (analysis)
  - Solution is specified in detail (design)
  - System that solves problem is built and installed (implementation)
  - System used, maintained, and enhanced to continue to provide intended benefits (support)

2

"Waterfall" Approach to the SDLC

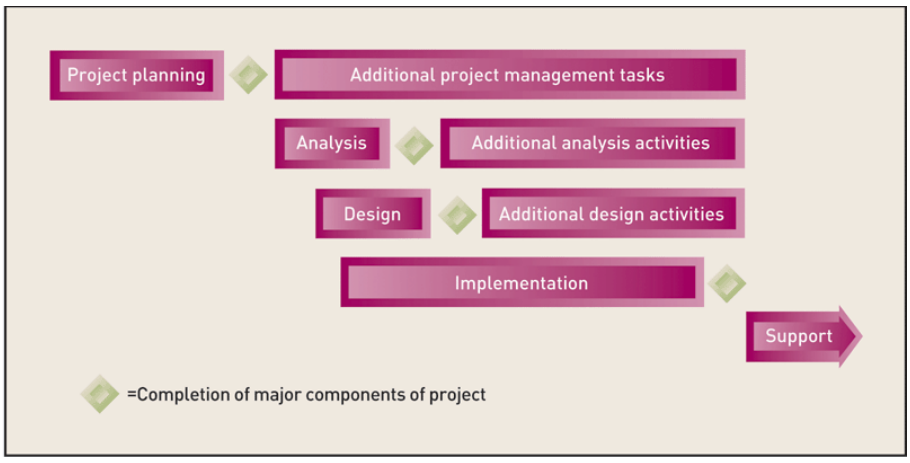
Figure 2-4

The waterfall approach to the SDLC



2

Modified Waterfall Approach with Overlapping Phases (Figure 2-5)



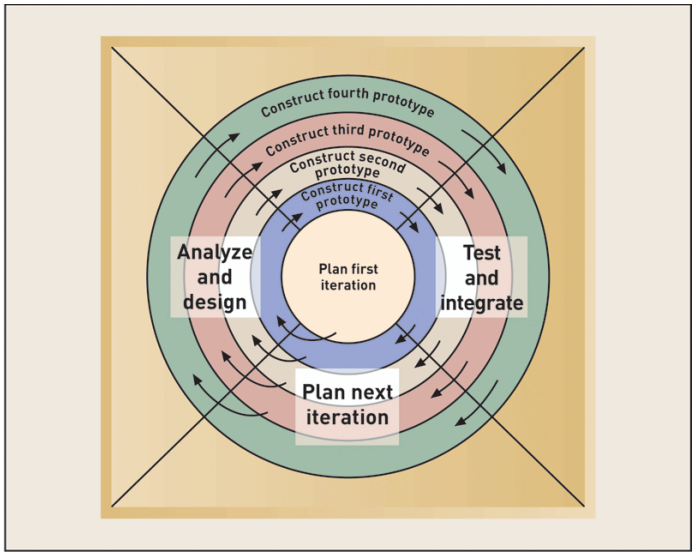
2

Newer Adaptive Approaches to the SDLC

- Based on spiral model
  - Project cycles through development activities over and over until project is complete
  - Prototype created by end of each cycle
  - Focuses on mitigating risk
- Iteration – Work activities are repeated
  - Each iteration refines previous result
  - Approach assumes no one gets it right the first time
  - There are a series of mini projects for each iteration

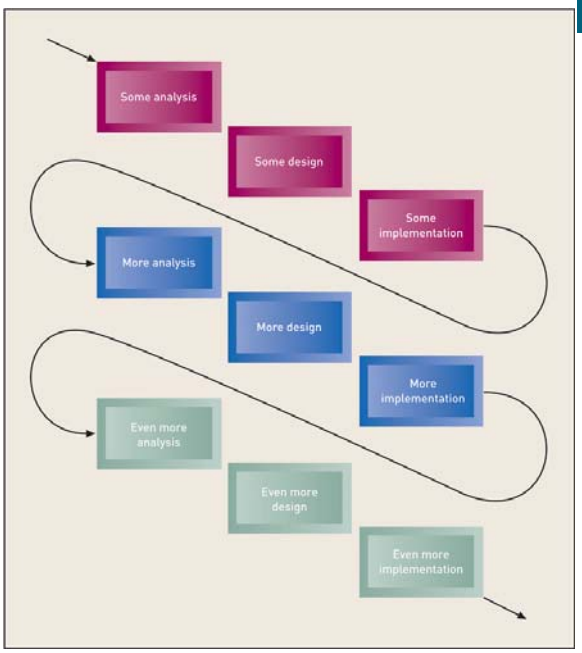
2

The Spiral Life Cycle Model (Figure 2-6)



2

Iteration of System Development Activities
(Figure 2-7)



11

2

Activities of Each SDLC Phase

- Predictive or adaptive approach use SDLC
- Activities of each “phase” are similar
- Phases are not always sequential
- Phases can overlap
- Activities across phases can be done within an iteration

12

2

Activities of Planning Phase of SDLC

- Define business problem and scope
- Produce detailed project schedule
- Confirm project feasibility
  - Economic, organizational, technical, resource, and schedule
- Staff the project (resource management)
- Launch project → official announcement

2

Activities of Analysis Phase of SDLC

- Gather information to learn problem domain
- Define system requirements
- Build prototypes for discovery of requirements
- Prioritize requirements
- Generate and evaluate alternatives
- Review recommendations with management

2

Activities of Design Phase of SDLC

- Design and integrate the network
- Design the application architecture
- Design the user interfaces
- Design the system interfaces
- Design and integrate the database
- Prototype for design details
- Design and integrate system controls

2

Activities of Implementation Phase of SDLC

- Construct software components
- Verify and test
- Convert data
- Train users and document the system
- Install the system

2

Activities of Support Phase of SDLC

- Maintain system
  - Small patches, repairs, and updates
- Enhance system
  - Small upgrades or enhancements to expand system capabilities
  - Larger enhancements may require separate development project
- Support users
  - Help desk and/or support team

2

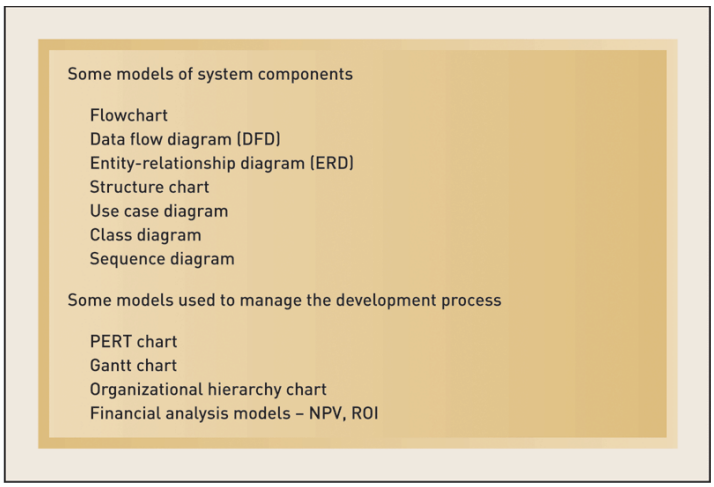
Methodologies and Models

- Methodologies
  - Comprehensive guidelines to follow for completing every SDLC activity
  - Collection of models, tools, and techniques
- Models
  - Representation of an important aspect of real world, but not same as real thing
  - Abstraction used to separate out aspect
  - Diagrams and charts
  - Project planning and budgeting aids

2

Some Models Used in System Development

Figure 2-8
Some models used in system development



2

Tools and Techniques

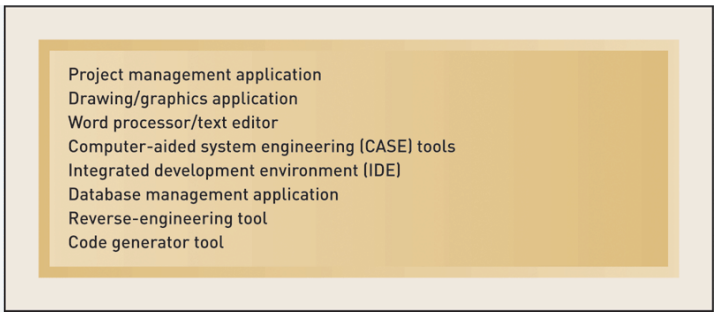
- Tools
  - Software support that helps create models or other required project components
  - Range from simple drawing programs to complex CASE tools to project management software
- Techniques
  - Collection of guidelines that help analysts complete a system development activity or task
  - Can be step-by-step instructions or just general advice

2

Some Tools Used in System Development

Figure 2-9

Some tools used in system development

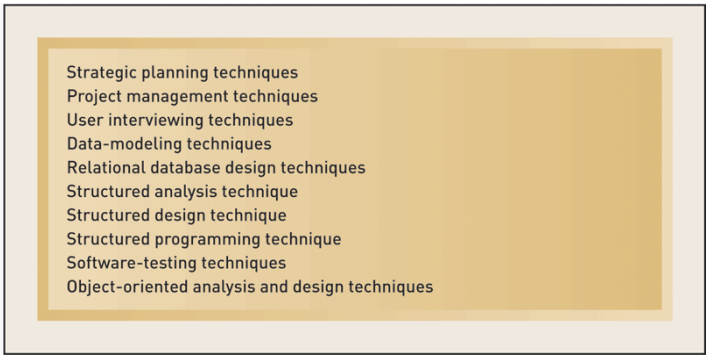


2

Some Techniques Used in System Development

Figure 2-10

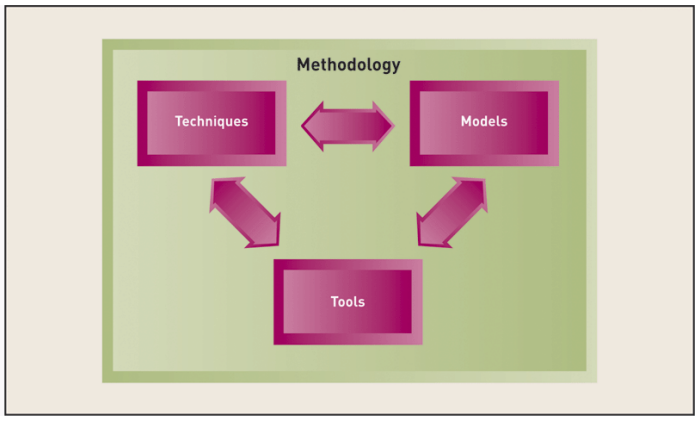
Some techniques used in system development



2

Relationships Among Components of a Methodology

Figure 2-11
Relationships among components of a methodology



2

Two Approaches to System Development

- Traditional approach
  - Also called structured system development
  - Structured analysis and design technique (SADT)
  - Includes information engineering (IE)
- Object-oriented approach
  - Also called OOA, OOD, and OOP
  - Views information system as collection of interacting objects that work together to accomplish tasks

2

Traditional Approach

- Structured programming
  - Improves computer program quality
  - Allows other programmers to easily read and modify code
  - Each program module has one beginning and one ending
  - Three programming constructs (sequence, decision, repetition)

2

Three Structured Programming Constructs

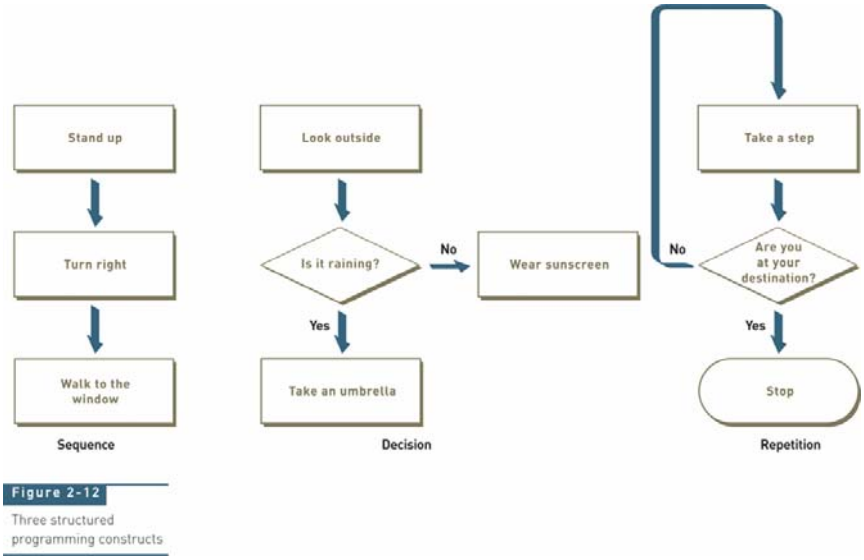


Figure 2-12
Three structured programming constructs

2

Top-Down Programming

- Divides complex programs into hierarchy of modules
- The module at top controls execution by “calling” lower level modules
- Modular programming
  - Similar to top-down programming
- One program calls other programs to work together as single system

27

2

Top-Down or Modular Programming

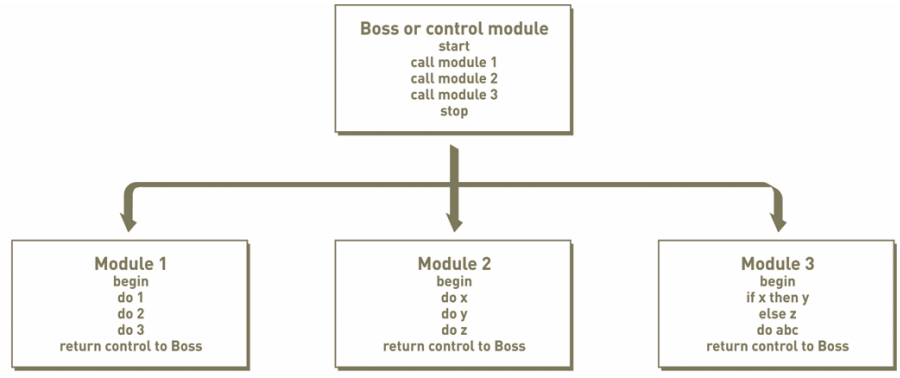


Figure 2-13
Top-down, or modular, programming

28

2

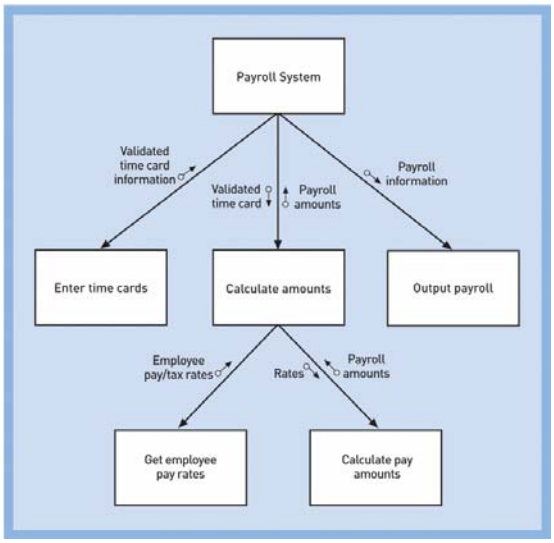
Structured Design

- Technique developed to provide design guidelines
  - What set of programs should be
  - What program should accomplish
  - How programs should be organized into a hierarchy
- Modules are shown with structure chart
- Main principle of program modules
  - Loosely coupled – module is independent of other modules
  - Highly cohesive – module has one clear task

2

Structure Chart Created Using Structured Design Technique

Figure 2-14
A structure chart created using the structured design technique



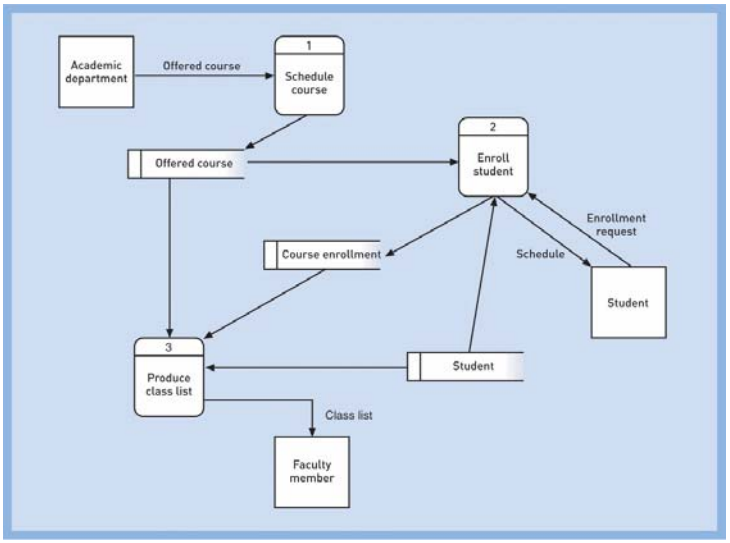
2

Structured Analysis

- Define what system needs to do (processing requirements)
- Define data system needs to store and use (data requirements)
- Define inputs and outputs
- Define how functions work together to accomplish tasks
- Data flow diagrams (DFD) and entity relationship diagrams (ERD) show results of structured analysis

2

Data Flow Diagram (DFD) Created Using Structured Analysis Technique (Figure 2-15)



Entity-Relationship Diagram (ERD) Created Using Structured Analysis Technique

2

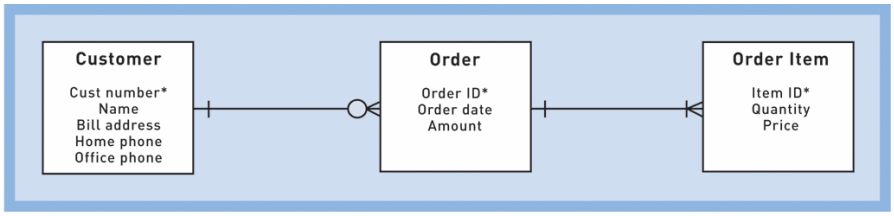
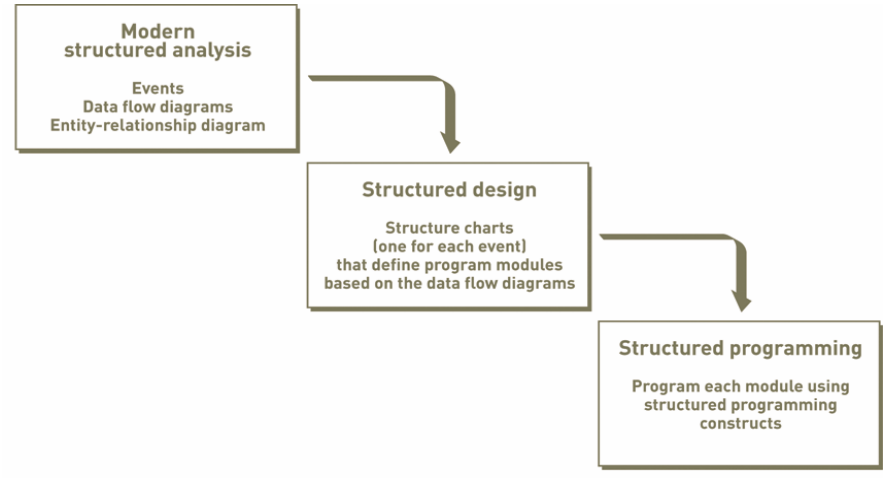


Figure 2-16

An entity-relationship diagram (ERD) created using the structured analysis technique

Structured Analysis Leads to Structured Design and Structured Programming (Figure 2-17)

2



2

Information Engineering (IE)

- Refinement to structured development
- Methodology with strategic planning, data modeling, automated tools focus
- More rigorous and complete than SADT
- Industry merged key concepts from structured development and information engineering approaches into traditional approach

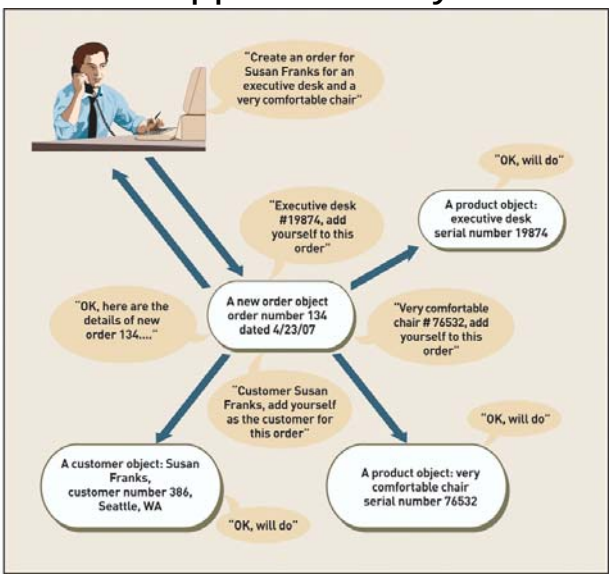
2

Object-Oriented Approach

- Completely different approach to information systems
- Views information system as collection of interacting objects that work together to accomplish tasks
  - Objects – things in computer system that can respond to messages
  - Conceptually, no processes, programs, data entities, or files are defined – just objects
- OO languages: Java, C++, C# .NET, VB .NET

Object-Oriented Approach to Systems

Figure 2-18
The object-oriented approach to systems (read clockwise starting with user)

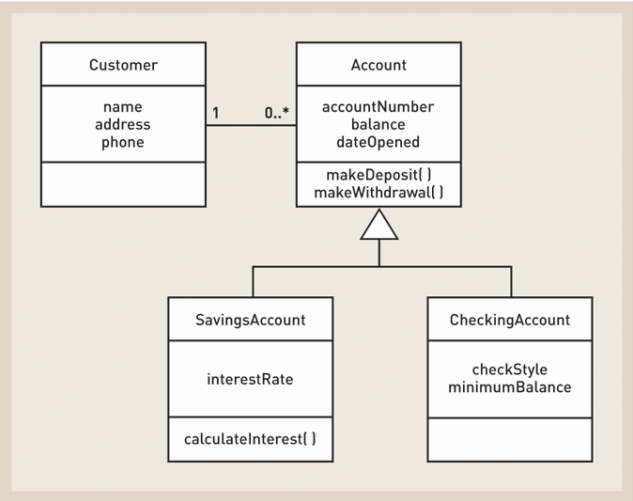


Object-Oriented Approach (continued)

- Object-oriented analysis (OOA)
  - Defines types of objects users deal with
  - Shows use cases are required to complete tasks
- Object-oriented design (OOD)
  - Defines object types needed to communicate with people and devices in system
  - Shows how objects interact to complete tasks
  - Refines each type of object for implementation with specific language of environment
- Object-oriented programming (OOP)
  - Writing statements in programming language to define what each type of object does

Class Diagram Created During OO Analysis

Figure 2-19
A class diagram created during object-oriented analysis

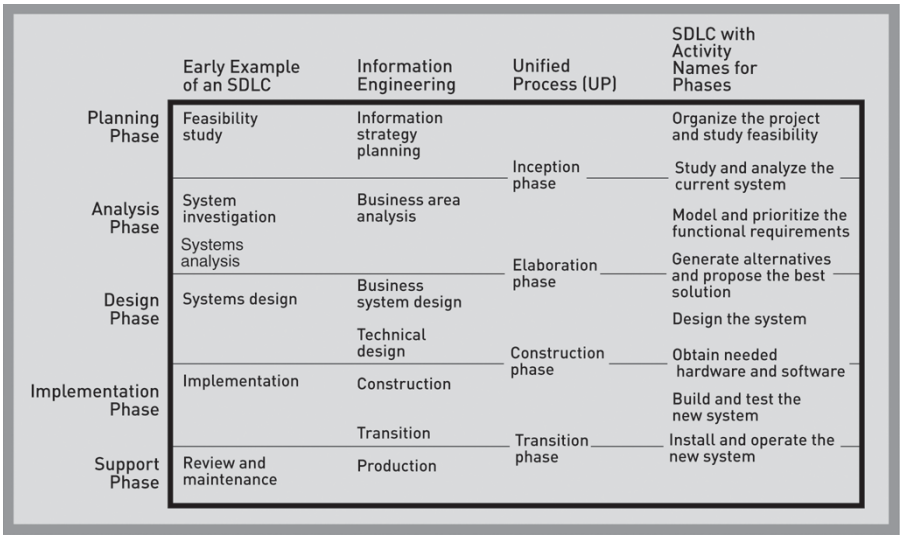


SDLC Variations

- Many variations of SDLC in practice
  - Based on variation of names for phases
  - No matter which one, activities/tasks are similar
- Some increase emphasis on people
  - User-centered design, participatory design
  - Sociotechnical systems
- Some increase speed of development
  - Rapid application development (RAD)
  - Prototyping

2

Life Cycles with Different Names for Phases (Figure 2-20)



2

Current Trends in Development

- More adaptive approaches
  - The Unified Process (UP)
  - Extreme Programming (XP)
  - Agile Modeling
  - Scrum
- Details on each in Chapter 16

2

The Unified Process (UP)

- Object-oriented development approach
- Offered by IBM / Rational
  - Booch, Rumbaugh, Jacobson
- Unified Modeling Language (UML) used primarily for modeling
- UML can be used with any OO methodology
- UP defines four life cycle phases
  - Inception, elaboration, construction, transition

2

The Unified Process (UP) (continued)

- Reinforces six best practices
  - Develop iteratively
  - Define and manage system requirements
  - Use component architectures
  - Create visual models
  - Verify quality
  - Control changes

2

Extreme Programming (XP)

- Recent, lightweight, development approach to keep process simple and efficient
- Describes system support needed and required system functionality through informal user stories
- Has users describe acceptance tests to demonstrate defined outcomes
- Relies on continuous testing and integration, heavy user involvement, programming done by small teams

2

Agile Modeling

- Hybrid of XP and UP (Scott Ambler); has more models than XP, fewer documents than UP
- Interactive and Incremental Modeling
  - Apply right models
  - Create several models in parallel
  - Model in small increments
- Teamwork
  - Get active stakeholder participation
  - Encourage collective ownership
  - Model with others and display models publicly

2

Agile Modeling (continued)

- Simplicity
  - Use simple content
  - Depict models simply
  - Use simplest modeling tools
- Validation
  - Consider testability
  - Prove model is right with code

2

Scrum

- For highly adaptive project needs
- Respond to situation as rapidly as possible
- Scrum refers to rugby game
  - Both are quick, agile, and self-organizing
- Team retains control over project
- Values individuals over processes

2

Tools to Support System Development

- Computer-aided system engineering (CASE)
  - Automated tools to improve the speed and quality of system development work
  - Contains database of information about system called repository
- Upper CASE – support for analysis and design
- Lower CASE – support for implementation
- ICASE – integrated CASE tools
- Now called visual modeling tools, integrated application development tools, and round-trip engineering tools

2

CASE Tool Repository Contains All System Information