Chapter 16:
Making the System Operational

Systems Analysis and Design in a Changing World, 3rd Edition

Learning Objectives

- Describe implementation and support activities
- Choose an appropriate approach to program development
- Describe various types of software tests and explain how and why each is used

Learning Objectives (continued)

- List various approaches to data conversion and system installation and describe the advantages and disadvantages of each
- Describe different types of documentation and the processes by which they are developed and maintained
- Describe training and user support requirements for new and operational systems

Overview

- This chapter focuses on activities of implementation and support phases of systems development life cycle (SDLC)
- Implementation activities occur before system is turned over to users
- Implementation consumes more time and resources than earlier phases of the SDLC
- Support activities occur after system becomes operational and may continue for years

Activities of the Implementation and Support Phases

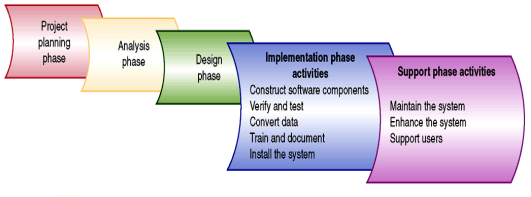


FIGURE 16-1
Activities of the implementation and support phases.

Program Development

- Program development is time consuming
  - One-third of development labor
  - One-third to one-half of project development schedule
- Programming and testing considerations:
  - Required resources
  - Managerial complexity
  - System quality

Order of Implementation

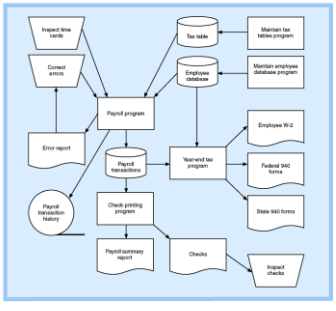
- Input, process, output (IPO) development order
  - Based on data flow through system
  - Simplifies testing
  - User interfaces developed early to reduce change
  - Disadvantage is late implementation of outputs
- Structured design – IPO order based on system flowchart and structure chart
- OO design – IPO order in package diagrams

Order of Implementation (continued)

- Top-down and bottom-up order from traditional structured design and structured programming
- Top-down begins with top structure chart module
  - Always a working version of program
  - Requires three or more iterations to complete
- Bottom-up begins with modules at lowest level of structure chart
  - Many programmers can begin immediately
  - Requires driver programs to test

System Flowchart for a Payroll System

FIGURE 16-2



Structure Chart for a Payroll System

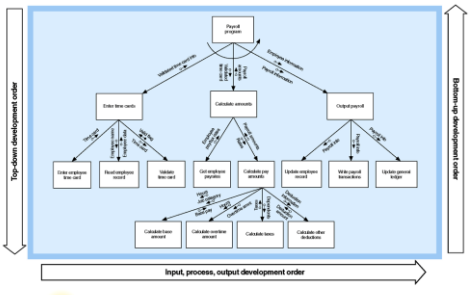


FIGURE 16-3

A structure chart for the Payroll program in Figure 16-2.

Package Diagrams for RMO Subsystems

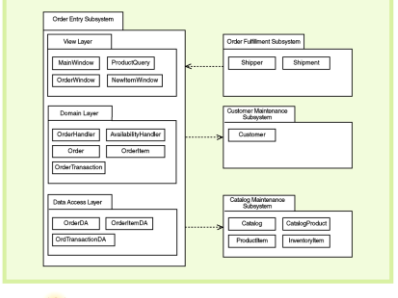


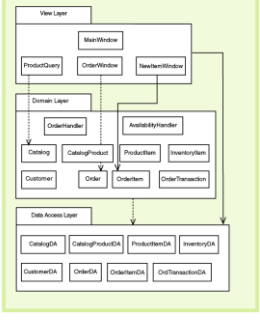
FIGURE 16-4

A package diagram for the four RMO subsystems.

Package Diagram for Three-Layer OO

FIGURE 16-5

A package diagram for a three-layer OO design.



Construction and Test Plan

- Development order
- Testing order
- Data used to test modules, module groups, methods, classes, programs, and subsystems
- Acceptance criteria
- Relevant personnel assignments (construction and testing)

Framework Development

- When developing large OO systems, object frameworks or foundation classes are often constructed
- Foundation classes typically implemented first
  - Minimizes impact of errors and changes
  - Reused in many parts of the system and across applications
  - Assigned to best programmers and thoroughly tested

Team-Based Program Development

- Management Issues
  - Organization of programming teams
  - Task assignment to specific teams or members
  - Member and team communication and coordination
- Variety of different models used for organization

Comparison and Summary of Development Team Types

FIGURE 16-6
A comparison and summary of development team types.

Team type	Team characteristics	Task and project types
Cooperating peers	Equal skill levels Overlapping specialities Consensus-based decision making	Experimentation Creative problem solving
Chief developer	Organized as a military platoon or squad One leader makes all important decisions	Well-defined objectives Well-defined path to completion
Collaborative specialists	Wide variation in skill and experience Minimal overlap in technical specialities Leader is primarily an administrator Consensus-based decision making	Diagnosis or experimentation Creative and integrative problem solving Wide range of technology

Source Code Control

- Source code control system (SCCS)
  - Automated tool for tracking source code files and controlling changes to those files
- Repository of code and programmer actions
  - Check out file in read-only mode
  - Check out file in read/write mode
  - Check in a modified file

Versioning

- Mechanism to manage systems changes
- Complex systems developed, installed, and maintained in series of versions to simplify testing and support
  - Alpha Version – incomplete testing version
  - Beta Version – end user testing version
  - Production Release Version – formally distributed to users or made operational
  - Maintenance Release – bug fixes, small changes

Description of Versions for RMO

FIGURE 16-9
Description of versions in Figure 16-8 for the RMO customer support system.

Alpha 0.1—Basic database functionality with simple CRUD capabilities. Handles regular transactions only, no reports or printing capability.

Alpha 0.2—Full CRUD for all transaction types with screens in near final form, no reports or printing capability. Includes bug fixes for version 0.1.

Beta 0.3—Screens in final form with simple on-line help, simple printing of screen contents. Includes bug fixes for version 0.2.

Beta 0.4—Adds reports and formatted printing. Includes bug fixes for version 0.3.

Production 1.0—Includes all bug fixes for version 0.4.

Production 1.1—Adds keystroke shortcuts for experienced users. Includes all bug fixes for version 1.0.

Alpha 1.9.1—Adds simple database extraction into a DSS tool for version 1.0.

Beta 1.9.2—Adds user-friendly database navigation and downloads into a DSS tool to version 1.1. Includes all bug fixes for version 1.9.1.

Production 2.0—Includes all bug fixes for version 1.9.2.

Quality Assurance

- Process of ensuring information system meets minimum quality standards
- Determined by users, implementation staff, management
- Identification of gaps or inconsistencies in systems requirements
- QA integrated into project throughout SDLC
- Cost of fixing errors rise as project progresses

Technical Reviews

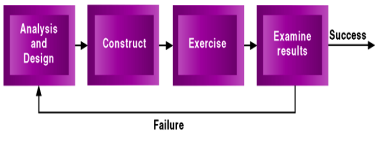
- Opens design and construction process to input from other people
- Other programmers can frequently see errors missed by original programmer
- Similar to author writing and editor reviewing
- Walkthroughs and inspections
  - Reduce number of errors by factor of 5 to 10
  - Reduce testing costs by 50%

Testing

- Process of examining a product to determine if any defects exist
- Testing levels are related to specific SDLC phases
- Testing activities spread throughout SDLC
- Most of testing takes place following software construction and definition of defect standards

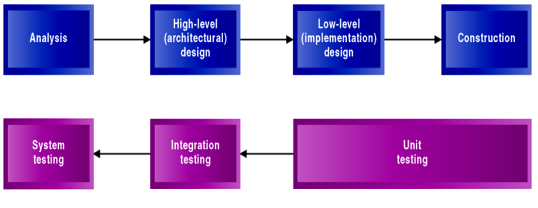
Generic Model of Software Testing

FIGURE 16-11
A generic model of software testing.



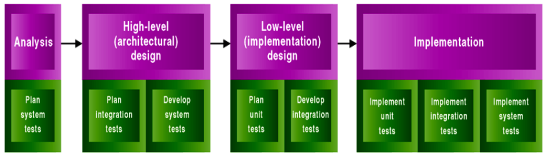
Correspondence Between SDLC Phases and Various Types of Testing

FIGURE 16-12 The correspondence between SDLC phases and various types of testing.



SDLC Phases and Testing Activities Performed Within Each Phase

FIGURE 16-13 SDLC phases and the testing activities that can be performed within each phase.



Test Cases

- Important part of testing is specifying test cases and data
- Test cases specify one or more events to which software must respond
  - Starting state
  - Events to which software responds
  - Expected response or ending state
- Analysis phase documentation is useful in preparing test cases

Unit Testing

- Testing individual modules of code or methods before integration with other software
- Driver module used for testing
  - Sets values of input parameters
  - Calls module to be tested and passes input parameters
  - Accepts return parameters from tested module
- Stub testing – test module simulates module not yet developed

Integration Testing

- Tests the behavior of a group of modules or methods
- Test both normal processing and exceptions
- Errors can include:
  - Interface incompatibility
  - Incorrect parameter values
  - Run-time exceptions
  - Unexpected state interactions

System Testing

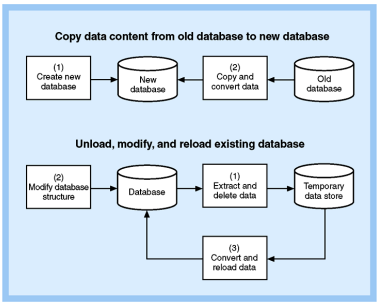
- Tests the behavior of the entire system
  - Build and smoke test is performed daily to discover any problems with daily builds
  - Performance test checks time-based requirements
  - Acceptance test is performed to determine whether system meets user requirements

Data Conversion

- Data needed at system startup
  - Files or databases of system being replaced
  - Manual records
  - Files or databases of other systems
  - User feedback during normal system operation
- Reuse of existing databases
- Reloading database contents
- Creating new databases

Two Approaches to Reloading Database Content After a Structural Modification

FIGURE 16-17
Two approaches to reloading database content after a structural modification.



A Complex Data-Conversion Example

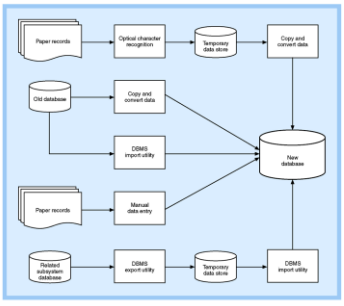


FIGURE 16-18

Installation

- After development and testing, system must be put into operation
- Important planning considerations
  - Costs of operating both systems in parallel
  - Detecting and correcting errors in new system
  - Potentially disrupting the company and IS operations
  - Training personnel and customers with new procedures

Direct Installation

- New system installed and quickly made operational
- Overlapping systems turned off
- Both systems concurrent for brief time
- Advantage: simplicity and fewer logistics issues to manage
- Disadvantage: risk due to no backup

Parallel Installation

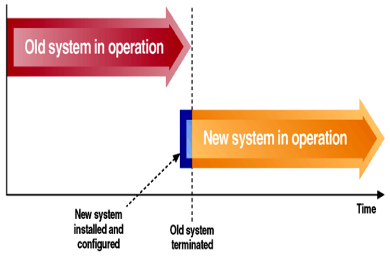
- Old and new systems operated together for extended period of time
- Advantages: low risk of system failure and continually backup
- Disadvantage: cost to operate both systems
  - Hiring temporary personnel
  - Acquiring extra space
  - Increasing managerial and logistical complexity

Phased Installation

- New system installed in series of steps or phases
- Each phase adds components to existing system
- Advantage: reduced risk because phase failure is less serious than system failure
- Disadvantage: multiple phases causes more activities, milestones, and management complexity for entire effort

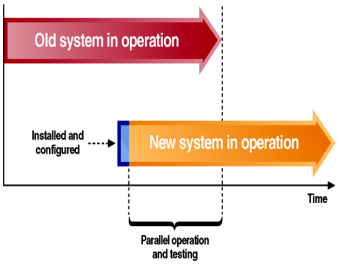
Direct Installation and Cutover

FIGURE 16-19
Direct installation and cutover.



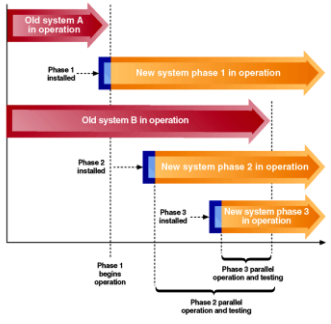
Parallel Installation and Operation

FIGURE 16-20
Parallel installation and operation.



Phased Installation with Direct Cutover and Parallel Operation

FIGURE 16-21
Phased installation with direct cutover and parallel operation.



Personnel Issues

- Installing new system places demands on personnel
  - Demanding schedules
  - Rapid learning and adaptation
  - High stress
- Planning should anticipate these risks and take measures to mitigate effects
- Temporary and contract personnel may be hired during an installation

Documentation

- Automated documentation is standard
  - Electronic manuals stored in MS Word or Adobe
  - Hyperlinked documents: Web browser formatted
  - On-line documentation on vendor Web site
  - Embedded documentation on CD
  - Electronic system model stored in graphic formats
  - Tool-specific system models developed with IDEs, DBMSs, and CASE tools

System Documentation

- Descriptions of system functions, architecture, and construction details
- Used by maintenance personnel and future developers
- Generated as a byproduct of development
  - Includes source code
  - Includes analysis and design models
- Failure to maintain system documentation compromises value of a system

Lifecycle Phases and System Documentation Generated in Each Phase

Life cycle phase	System documentation	
	Traditional approach	Object-oriented approach
Analysis	Entity-relationship diagram Data flow diagram Process description Data flow and element definition	Class diagram Use case Activity diagram System sequence diagram
	Event list	
Design	System flowchart Structure chart	Design class diagram Interaction diagram Collaboration diagram Package diagram Statechart
	Module or method pseudocode Database schema diagram	
Implementation	Program source code Database schema source code Test data	

FIGURE 16-22
Life cycle phases and system documentation generated in each phase.

User Documentation

- Descriptions of how to interact with and maintain the system
- Used by end users and system operators
- Topics covered include:
  - Startup and shutdown
  - Keystrokes, mouse, or command functions to perform specific functions
  - Program function for specific business procedures
  - Common errors and correction techniques

Training and User Support

- Without training, user error rates will be high
- Training considerations
  - Frequency and duration of system use
  - Need to understand system's business context
  - Existing computer skills and proficiency
  - Number of users

Typical Activities of End Users and Systems Operators

FIGURE 16-24
Typical activities of end users and system operators.

End user activities	System operator activities
Creating records or transactions	Starting or stopping the system
Modifying database contents	Querying system status
Generating reports	Backing up data to archive
Querying database	Recovering data from archive
Importing or exporting data	Installing or upgrading software

Ongoing Training and User Support

- User support covers training and user assistance that occurs after installation
  - On-line documentation and troubleshooting
  - Resident experts
  - Help desk
  - Technical support

Maintenance and System Enhancement

- Modification of software after delivery to correct faults, improve performance, or adapt the product to a changed environment
  - Tracking modification requests and changes
  - Implementing changes
  - Monitoring system performance
  - Upgrading hardware/software
  - Updating documentation

Submitting Change Requests and Error Reports

- Most organizations adopt formal change control procedures to manage change risks
  - Standard change request forms
  - Review of requests by change control committee
  - Extensive planning for design and implementation
- Approved changes are added to list of pending changes for budgeting, scheduling, planning, and implementation
- A separate process is used for error correction

Implementing a Change

- Planning for a change includes:
  - Identify parts of system to change or addition
  - Secure personnel to implement change
  - Schedule design and implementation activities
  - Develop test criteria and testing plan for changed system
- System documentation is reviewed to determine scope of change

Upgrading Computing Infrastructure

- Infrastructure requires periodic updates
  - Software maintenance releases
  - Software version upgrades
  - Declining system performance
- Infrastructure includes computer hardware, system software, networks, DBMSs
  - Technical, complex, and risky
  - Outages can impact entire system

Summary

- Implementation activities occur after design and before system is turned over to users
- Implementation is complex
  - Interdependence of programming, quality assurance, hardware and software installation, documentation and training
- Implementation is difficult to manage
  - Activities must be properly sequenced
  - Progress must be continually monitored

Summary (continued)

- Implementation is risky
  - Significant time and resources required
  - Often affects systems vital to daily operations
- Software components constructed in order to:
  - Minimize development resources needed
  - Maximize ability to test system and control errors
  - These goals often conflict: trade-off among resources, time, and desire to correct errors

Summary (continued)

- Data conversion, installation, documentation, and training follow programming and testing
- Installed and documented system is prerequisite for complete training
- Fully populated database needed to begin operation
- Support activities occur after system becomes operational and may continue for years to support user requirements and reduce operational risk