Department of Computer & Information Sciences
Pakistan Institute of Engineering and Applied Sciences

The Systems Development Environment

Chapter1

(Slides based on Prentice Hall/Pearson Resource Material for Book) Umar Faiz

Modern Systems Analysis and Design

Fifth Edition

Jeffrey A. Hoffer
Joey F. George
Joseph S. Valacich

Chapter 1

The Systems Development Environment

Chapter 1

The Systems Development Environment

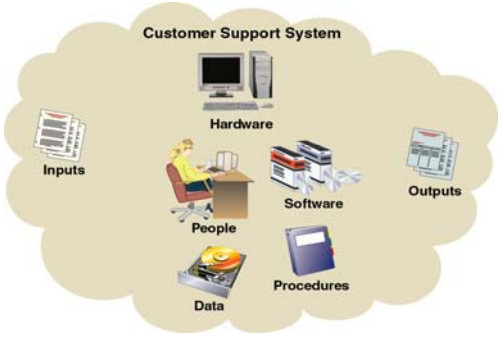
Learning Goals

1. Define information systems analysis and design.
2. Describe the different types of information systems.
3. Describe the information Systems Development Life Cycle (SDLC).
4. Explain Rapid Application Development (RAD), prototyping, Joint Application Development (JAD), and Computer Aided Software Engineering (CASE).
5. Describe agile methodologies and eXtreme programming. Explain the importance of special network technologies.
6. Explain Object Oriented Analysis and Design and the Rational Unified Process (RUP).

Information Systems

- Why Do People Need Information?
  - Individuals - Entertainment and enlightenment
  - Businesses - Decision making, problem solving and control

Information Systems

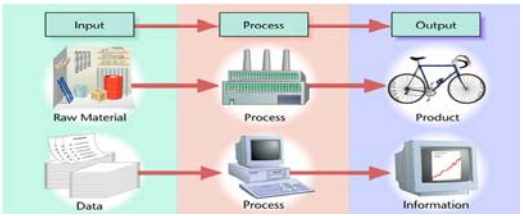


Data, Information, and Systems

- Data vs. Information
  - Data are raw facts about the organization and its business transactions. Most data items have little meaning and use by themselves.
  - Information is data that has been refined and organized by processing and purposeful intelligence.

Data, Information, and Systems

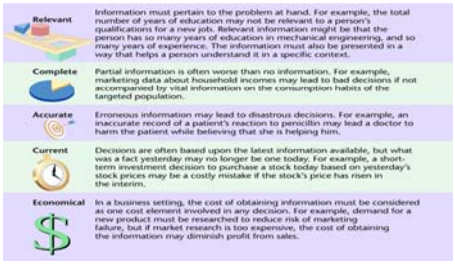
- Generating Information
  - CB Information systems take data as raw material, process it, and produce information as output.



(*Principles of Information Systems* by Stair and Reynolds; Fifth Edition)

Data, Information, and Systems

- Information in Context

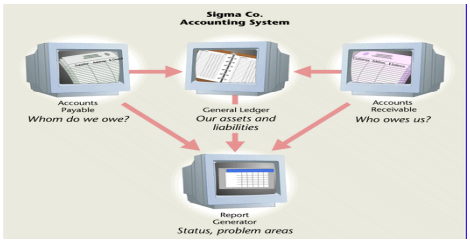


(*Principles of Information Systems* by Stair and Reynolds; Fifth Edition)

Data, Information, and Systems

- What Is a System?
  - System: A set of components that work together to achieve a common goal
  - Subsystem: One part of a system where the products of more than one system are combined to reach an ultimate goal
  - Closed system: Stand-alone system that has no contact with other systems
  - Open system: System that interfaces with other systems

Data, Information, and Systems



Several subsystems make up this corporate accounting system (*Principles of Information Systems* by Stair and Reynolds; Fifth Edition)

Data, Information, and Systems

- Information and Managers
  - Systems thinking
    - Creates a framework for problem solving and decision making.
    - Keeps managers focused on overall goals and operations of business.

Data, Information, and Systems

- Humans VS Computers
  - Humans
    - Think
    - Have common sense
    - Can make decisions
    - Can instruct the computer what to do
    - Can learn new methods and techniques
    - Can accumulate expertise

Data, Information, and Systems

- Humans VS Computers
  - Computers
    - Calculate and perform logical operations efficiently
    - Store and retrieve data and information extremely rapidly
    - Perform complex logical and arithmetical functions accurately
    - Execute tasks in lesser time
    - Reprogrammable

13

Data, Information, and Systems

- The Benefits of Human-Computer Synergy
  - Synergy
    - When combined resources produce output that exceeds the sum of the outputs of the same resources employed separately
  - Facilitates human thought process to be translated in terms of processing of large amounts of data efficiently

14

Data, Information, and Systems

- Components if an Information System

Data	Input that the system takes to produce information.
Hardware	A computer and its peripheral equipment: input, output, and storage devices. Hardware also includes data communication equipment.
Software	Sets of instructions that tell the computer how to take data in, how to process it, how to display information, and how to store data and information.
Telecommunications	Hardware and software that facilitate fast transmission and reception of text, pictures, sound, and animation in the form of electronic data.
People	Information systems professionals and users who analyze organizational information needs, design and construct information systems, write computer programs, operate the hardware, and maintain software.
Procedures	Rules for achieving optimal and secure operations in data processing. Procedures include priorities in running different applications on the computer and security measures.

(*Principles of Information Systems* by Stair and Reynolds; Fifth Edition)

15

Data, Information, and Systems

- The Four Stages of Data Processing
  - Input: Data is collected and entered into computer.
  - Data processing: Data is manipulated into information using mathematical, statistical, and other tools.
  - Output: Information is displayed or presented.
  - Storage: Data and information are maintained for later use.

16

Why Study IS?

- Information Systems Careers
  - Systems analyst, specialist in enterprise resource planning (ERP), database administrator, consulting, etc.
- Knowledge Workers
  - Managers and non-managers
  - Employers seek computer-literate professionals who know how to use information technology.

17

Ethical and Societal Issues
The Not-So-Bright Side

- Consumer Privacy
  - Organizations collect (and sometimes sell) huge amounts of data on individuals.
- Employee Privacy
  - IT supports remote monitoring of employees, violating privacy and creating stress.

18

Information Systems & Technology

- An information system (IS) is an arrangement of people, data, processes, communications, and information technology that interact to support and improve day-to-day operations in a business as well as support the problem-solving and decision making needs of management and users.

Information Systems & Technology

- Information technology is a contemporary term that describes the combination of computer technology (hardware and software) with telecommunications technology (data, image, and voice networks).

Information Systems in Organizations

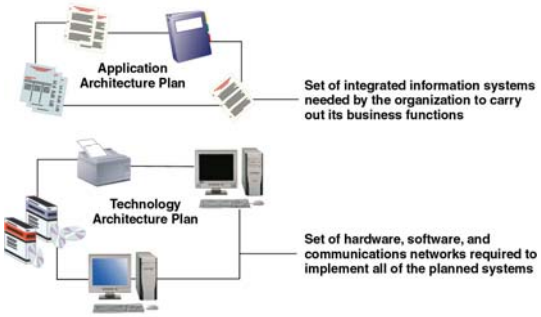
- Enhanced speed and quantity of information available improves the speed and effectiveness of decision making.
- Computers make accurate, unbiased data available to everyone.
- Information-sharing capabilities support team decision making at low levels of an organization's hierarchy.
- Users to collaborate on reports and other projects from different locations.

Information Systems Strategic Planning

- Purpose: anticipate problems
- Large plan comprised of models and smaller plans
  - Organization model: maps business functions
  - Application architecture plan: lists integrated information systems
  - Technology architecture plan: defines hardware, software, and communications networks

22

Information Systems Strategic Plan



Components of an Information Systems Strategic Plan

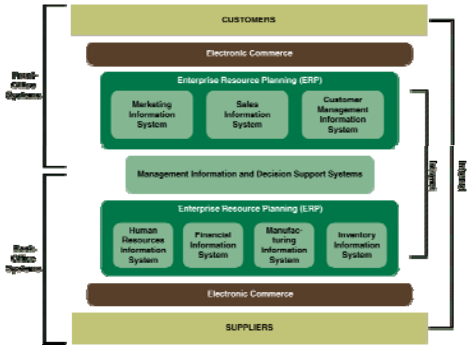
Front- and Back-Office Information Systems

- Front-office information systems support business functions that reach out to customers (or constituents).
  - Marketing
  - Sales
  - Customer management

Front- and Back-Office Information Systems

- Back-office information systems support internal business operations and interact with suppliers (of materials, equipment, supplies, and services).
  - Human resources
  - Financial management
  - Manufacturing
  - Inventory control

A Federation of Information Systems



Information Systems in Organizations

- Information Systems Analysis and Design
  - Complex organizational process.
  - Used to develop and maintain computer-based information systems.
  - Used by a team of business and systems professionals.
- An important result of systems analysis and design is the Application Software

27

Application Software

- Application software
  - A program that performs the specific tasks that the user wants to carry out.
  - Examples: Oracle Supply Chain Management Suite, Microsoft Excel, Adobe Acrobat, and QuickBooks.

Organizational Approach to System Analysis

- An organizational approach to systems analysis and design is driven by
  - Methodologies
  - Techniques
  - Tools

Organizational Approach to System Analysis

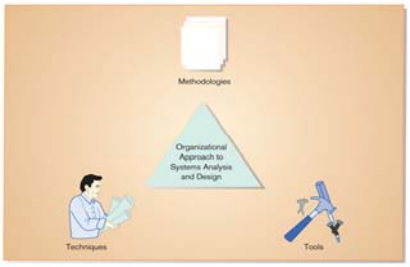


Figure 1-1 An organizational approach to systems analysis and design is driven by methodologies, techniques, and tools

Organizational Approach to System Analysis

- Methodologies
  - Methodologies are comprehensive, multiple-step approaches to systems development that will guide you r work and influence the quality of information system.
- Techniques
  - These are particular processes that help ensure that the work undertaken is is well thought-out, complete, and comprehensible to others

Organizational Approach to System Analysis

- Tools
  - These are programs that benefit from the techniques and execute the steps required for overall methodology

Modern Approach to System Analysis

- A Modern Approach to Systems Analysis and Design
  - 1950s: focus on efficient automation of existing processes
  - 1960s: advent of 3GL, faster and more reliable computers
  - 1970s: system development becomes more like an engineering discipline

33

Modern Approach to System Analysis

- A Modern Approach to Systems Analysis and Design
  - 1980s: major breakthrough with 4GL, CASE tools, object oriented methods
  - 1990s: focus on system integration, client/server platforms, Internet
  - The new century: Web application development, wireless PDAs, component-based applications

34

Stakeholders in Systems Development

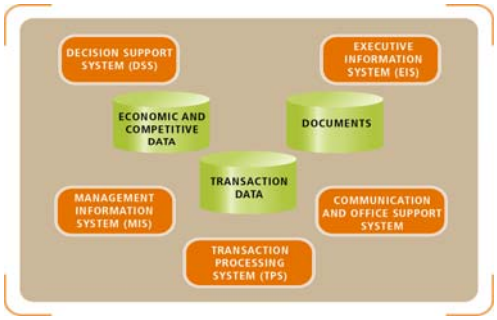
- IS Managers
- Systems Analyst
  - Organizational role most responsible for analysis and design of information systems.
- Programmer
- Business Managers

35

Classes of Information Systems

- Transaction Processing Systems (TPS)
- Customer Integrated Systems (CIS)
- Management Information Systems (MIS)
- Decision Support Systems
- Executive Information Systems
- Expert Systems
- Office Automation Systems
  - Communication and Office Support Systems

Types of Information Systems



37

Types of Information Systems

- Transaction Processing Systems (TPS)
  - A Transaction Processing System (TPS) is a type of information system that collects, stores, modifies and retrieves the data transactions of an enterprise.
    - Example: Airline Reservation System
  - **Business process redesign** (BPR) is the study, analysis, and redesign of fundamental business (transaction) processes to reduce costs and/or improve value added to the business.

Types of Information Systems

- Transaction Processing Systems (TPS)
  - A Transaction Processing System (TPS) is a type of information system that collects, stores, modifies and retrieves the data transactions of an enterprise.
  - Example: Airline Reservation System

39

Types of Information Systems

- TPS Requirements
  - Capture, process and store transactions
  - Produce reports and information about transactions
  - Produce transactions for other TPS
  - Be accurate and timely
  - Be efficient – require less labor
  - Increase customer service
  - Increase competitive advantage

40

Types of Information Systems

- TPS Activity Model
  - Data capture
    - From online entry or "external" system
  - Data validation
    - Should be done as close to source as possible
  - Processing
    - Data manipulation
    - Database updating
    - Generation of transactions for other systems
  - Archive Transaction
  - Create documents and reports

41

Types of Information Systems

- Customer Integrated Systems
  - Customer integrated systems (CIS), or customer self-service systems, is an extension of the transaction processing system that places technology in the hands of an organization's customers and allows them to process their own transactions.

Types of Information Systems

- Management Information Systems (MIS)
  - A **management information system** (MIS) is an information system application that provides for management-oriented reporting. These reports are usually generated on a predetermined schedule and appear in a prearranged format.
    - Converts raw data from transaction processing system into meaningful form (Data orientation)

43

Types of Information Systems

- Decision Support Systems
  - A decision support system (DSS) is an information system application that provides its users with decision-oriented information whenever a decision-making situation arises. When applied to executive managers, these systems are sometimes called executive information systems (EIS).

44

Types of Information Systems

- Decision Support System
  - A data warehouse is a read-only, informational database that is populated with detailed, summary, and exception data and information generated by other transaction and management information systems. The data warehouse can then be accessed by end-users and managers with DSS tools that generate a virtually limitless variety of information in support of unstructured decisions.

45

Types of Information Systems

- Executive Information Systems
  - EIS are an extension of DSS. When applied to executive managers, these systems are sometimes called executive information systems (EIS).

46

Types of Information Systems

- Expert Systems
  - An expert system is a programmed decision-making information system that captures and reproduces the knowledge and expertise of an expert problem solver or decision maker and then simulates the “thinking” or “actions” of that expert.
    - Expert systems are implemented with artificial intelligence technology that captures, stores, and provides access to the reasoning of the experts.

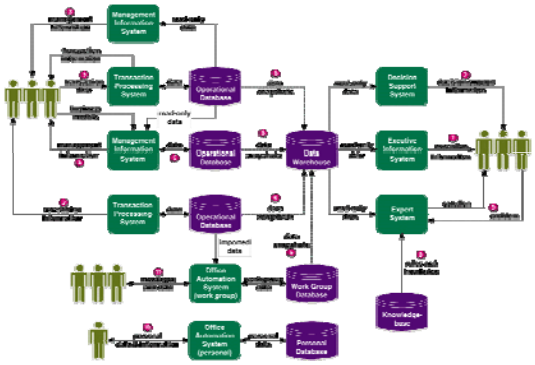
Types of Information Systems

- Office Automation Systems
  - Office automation (OA) systems support the wide range of business office activities that provide for improved work flow and communications between workers, regardless of whether or not those workers are located in the same office.

Types of Information Systems

- Office Automation Systems
  - Personal information systems are those designed to meet the needs of a single user. They are designed to boost an individual's productivity.
  - Work group information systems are those designed to meet the needs of a work group. They are designed to boost the group's productivity.

Information Systems Applications



Life Cycle versus Methodology

- A system life cycle divides the life of an information system into two stages, systems development and systems operation and support.

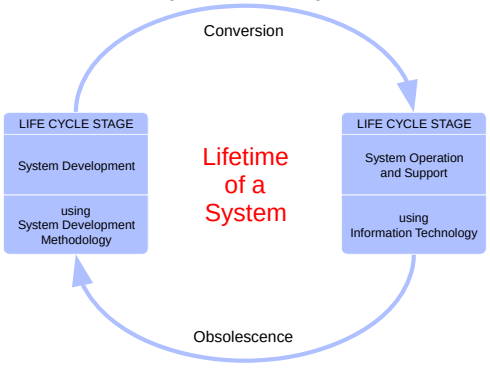
Process of System Development

- A system development process is a set of activities, methods, best practices, deliverables, and automated tools that stakeholders use to develop and maintain information systems and software.

Life Cycle versus Methodology

- A system development methodology is a very formal and precise system development process that defines a set of activities, methods, best practices, deliverables, and automated tools that system developers and project managers are to use to develop and maintain information systems and software.

A System Life Cycle



Principles of System Development

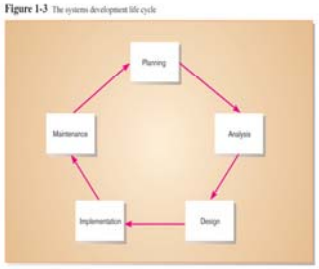
- Get the owners and users involved.
- Use a problem-solving approach.
- Establish phases and activities.
- Establish standards for development and documentation.
- Justify systems as capital investments.
- Don't be afraid to cancel or revise scope.
- Divide and conquer.
- Design systems for growth and change.
- Proper planning and project mgmt

Systems Development Life Cycle (SDLC)

- Traditional methodology used to develop, maintain, and replace information systems.
- Phases in SDLC:
  - Planning
  - Analysis
  - Design
  - Implementation
  - Maintenance

56

Standard and Evolutionary Views of SDLC



57

Standard and Evolutionary Views of SDLC

- Standard and Evolutionary Views of SDLC

58

Systems Development Life Cycle

- Planning
  - An organization's total information system needs are identified, analyzed, prioritized, and arranged.
- Analysis
  - System requirements are studied and structured.

59

Systems Development Life Cycle

- Design
  - A description of the recommended solution is converted into logical and then physical system specifications.
- Logical design
  - All functional features of the system chosen for development in analysis are described independently of any computer platform.

60

Systems Development Life Cycle

- Physical design
  - The logical specifications of the system from logical design are transformed into the technology-specific details from which all programming and system construction can be accomplished.

61

Systems Development Life Cycle

- Implementation
  - The information system is coded, tested, installed and supported in the organization.
- Maintenance
  - An information system is systematically repaired and improved.

62

Table 1-2 Products of SDLC Phases

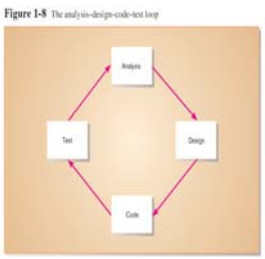
Phase	*Products, Outputs, or Deliverables*
Planning	Priorities for systems and projects; an architecture for data, networks, and selection hardware, and IS management are the result of associated systems;
	Detailed steps, or work plan, for project;
	Specification of system scope and planning and high-level system requirements or features;
	Assignment of team members and other resources;
	System justification or business case
Analysis	Description of current system and where problems or opportunities are with a general recommendation on how to fix, enhance, or replace current system;
	Explanation of alternative systems and justification for chosen alternative
Design	Functional, detailed specifications of all system elements (data, processes, inputs, and outputs);
	Technical, detailed specifications of all system elements (programs, files, network, system software, etc.);
	Acquisition plan for new technology
Implementation	Code, documentation, training procedures, and support capabilities
Maintenance	New versions or releases of software with associated updates to documentation, training, and support

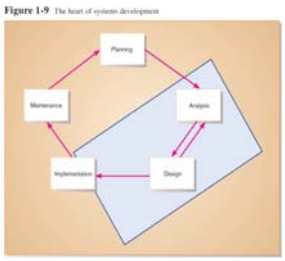
Systems Development Process/Model

- Traditional
- Alternatives
  - Prototyping
  - Rapid Application Development
  - CASE-based
  - Joint Application Development

64

Systems Development Model/Process





Current practice combines analysis, design, and implementation into a single iterative and parallel process of activities

65

Traditional Waterfall SDLC

- From four to 12 phases
- One phase completed before beginning next phase
- Emphasis on documentation and checkpoints
- Detailed planning and budgeting at each phase

66

Stages of SDLC

- Phase and Deliverable
  - Pre-investigation
    - Project proposal and recommendation
  - Investigation
    - Investigation Report
    - Analysis of alternatives
    - Cost and benefit analysis
    - Schedule
    - Recommendation

67

Stages of SDLC

- Phase and Deliverable
  - Requirements Analysis
    - Detailed system requirements
    - Revised cost and benefits analysis
    - Revised schedule

68

Stages of SDLC

- Phase and Deliverable
  - Design
    - Logical design
    - Technical design
    - Test plans
    - Revised cost and benefits analysis
    - Revised schedule

69

Stages of SDLC

- Phase and Deliverable
  - Construction & Testing
    - System elements,
    - Implementation plans
    - Contingency plans
    - End user training materials
    - Maintenance plan
    - Operations plans

70

Stages of SDLC

- Phase and Deliverable
  - Implementation
    - Cutover / Phase in

71

Pros and Cons of SDLC

- Advantages
  - Lends itself to good control
  - Phase deliverables well defined
    - Facilitates contracting
  - Clear checkpoints makes reviews easy
  - Creates detailed documentation which is valuable for maintenance

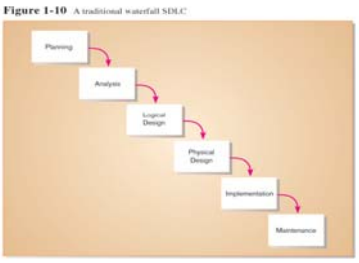
72

Pros and Cons of SDLC

- Disadvantages
  - Time and cost estimation difficult
  - Can be very slow
  - System requirements “locked in” after being determined (can't change).
  - Limited user involvement (only in requirements phase).

73

Traditional Waterfall SDLC



One phase begins when another completes, little backtracking and looping

74

Alternative Approaches to Systems Development

- Prototyping
- Computer-Aided Software Engineering (CASE) Tools
- Joint Application Design (JAD)
- Rapid Application Development (RAD)
- Agile Methodologies
- eXtreme Programming

75

Prototyping

- Requirements quickly converted to a working system (or a portion of it)
- Close collaboration between users and analysts.
- Iterative development process, system is continually revised.

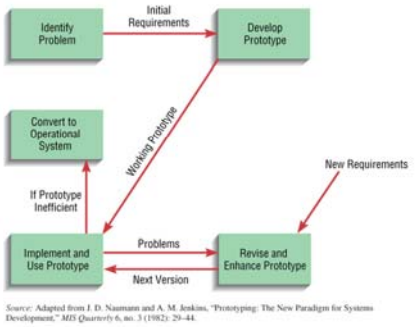
76

Prototyping

- Expendable (throw-away) prototyping:
  - Mockups that don’t really work, discarded after use
  - Used to support the analysis and design phases
  - Limited to input and output (reports, screens)
- Evolutionary prototyping:
  - Prototype evolves into the final system
  - Working systems or portions of systems
  - Possibly 4GL based

77

Prototyping



Source: Adapted from J. D. Naumann and A. M. Jenkins, “Prototyping: The New Paradigm for Systems Development,” *MIS Quarterly* 6, no. 3 (1982): 29–44.

78

Prototyping

- Advantages
  - When there is uncertainty about requirements or design solutions
    - can capture requirements in concrete, rather than verbal or abstract form
    - users are more likely to be able to state their detailed requirements when they see and use a prototype
    - users are more likely to get what they want

Prototyping

- Advantages
  - when there are several stakeholders
    - convenient display method for multiple parties
  - because it encourages user participation
    - user can relay feedback immediately
    - changes can be made interactively
  - because it is easier to identify behavioural issues when users are using the prototype
    - the designer can interactively accommodate the way the user 'uses' the interface

Prototyping

- Disadvantages
  - tends to skip through analysis and design phases too quickly --> lack of thorough understanding of the problems
    - a tendency to avoid creating formal documentation of system requirements which can then make the system more difficult to develop into a production system
  - can discourage consideration of a wide range on alternative design options .. tendency to go with the first one that the user likes

Prototyping

- Disadvantages
  - often lacks flexibility, technical efficiency and maintainability because of hasty construction
  - not suitable for large applications which have large amounts of data and multiple users - hard to control
  - often built as stand-alone systems, thus ignoring issues of data sharing and interactions with other existing systems

Prototyping

- Disadvantages
  - checks in the SDLC are bypassed so tendency to gloss over essential tasks eg. feasibility, standardisation, documentation, testing, security, etc..
  - can become too specific to the user representative and difficult to adapt to other potential users

Joint Application Development (JAD)

- Is actually analysis and design
  - Originated in late 1970s at IBM
- Brings together key users, managers, systems analysts in a group interview with a specific structure of roles and agenda
- Purpose
  - collect key system requirements
  - develop system design

Joint Application Development (JAD)

- Group meeting
  - formal agenda
  - avoid distractions
  - identify areas of agreement and conflict
  - resolve conflicts during the period of sessions
  - focus on rapid delivery of analysis and design specifications

Joint Application Development (JAD)

- JAD participants
  - facilitator - organises and runs the sessions
  - scribe(s) - takes notes on PC, CASE tool etc
  - users - understand the system requirements
  - managers - organisational overview
  - systems analysts - technical knowledge, learn about the system
  - sponsor - senior executive who commits and funds the process

Joint Application Development (JAD)

- JAD sessions
  - from one to five days
  - structured meeting room (war room) with white boards, CASE tools etc
  - located away from users' workplace
  - outcome is documents detailing the system - workings of/requirements for the system, system design specifications, prototypes

Joint Application Development (JAD)

- Preparing for JAD sessions
  - JAD leader prepares and distributes agenda and documentation about scope and objectives
  - Agenda specifies issues to be discussed and time allocated to each
  - Ground rules for running the sessions are made clear
  - Ensure users who attend are knowledgeable about their business area

Joint Application Development (JAD)

- Conducting JAD sessions:
  - Avoid deviating from the agenda
  - Keep to schedule (time for topics)
  - Ensure scribe takes adequate notes
    - use formal minutes
  - Avoid using technical jargon
    - involve all participants
  - Use conflict resolution strategies

Joint Application Development (JAD)

- Conducting JAD sessions:
  - Allow ample breaks
    - keep everyone at peak efficiency
  - Encourage group consensus
  - Encourage participation vs individuals dominating
  - Ensure ground rules are adhered to

CASE Tools

- CASE
  - CASE can be defined as software tools that provide automated support for some portion of the systems development process.
  - It helps to enact a single design philosophy within an organization with many projects, systems and people.

CASE Tools

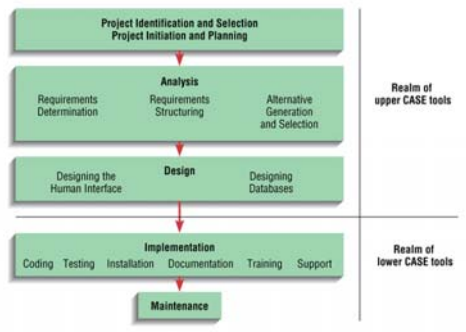
- CASE tools
  - Helps to provide an engineering-type discipline to software development
  - Automation of the entire software life cycle process.
  - Assists the system builders in managing the complexities of information system projects
  - Helps assure that high-quality systems are constructed on time and within budget.

CASE Tools

- CASE can be defined as software tools that provide automated support for some portion of the systems development process.

CASE Tools

- Upper CASE tools are designed to support the information planning and the project identification and selection, project initiation and planning, analysis, and design phases of the systems development life cycle.
- Lower CASE tools support the implementation and maintenance phases of the systems development life cycle.



CASE Tools

- Cross life-cycle CASE tools are designed to support activities that occur across multiple phases of the systems development life cycle.

CASE Tools

- I-CASE (Integrated CASE) refers to an automated systems development environment that provides numerous tools to create diagrams, forms, and reports; provides analysis, reporting, and code generation facilities; and seamlessly shares and integrates data across and between tools.

CASE Tools

- Diagramming tools enable graphical representation.
- Computer displays and report generators help prototype how systems "look and feel".
- Analysis tools automatically check for consistency in diagrams, forms, and reports.
- Central repository for integrated storage of diagrams, reports, and project management specifications.

98

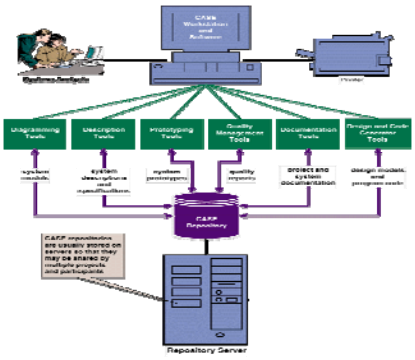
CASE Tools

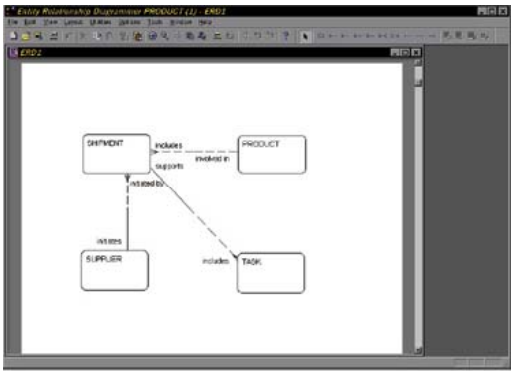
- Documentation generators standardize technical and user documentation.
- Code generators enable automatic generation of programs and database code directly from design documents, diagrams, forms, and reports.

99

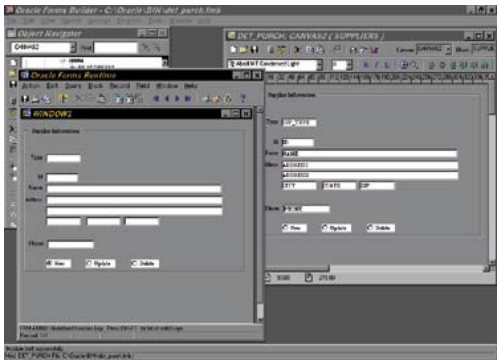
The general types of CASE tools

CASE Architecture

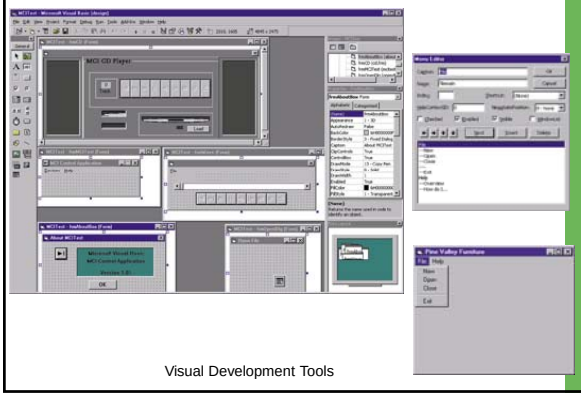




Diagramming tools

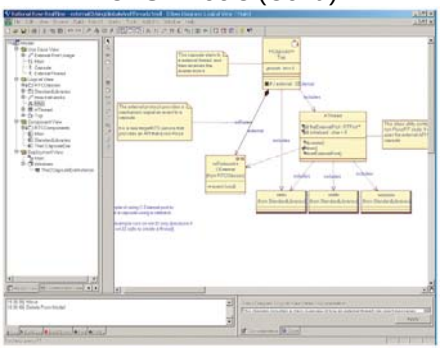


Computer display and report generators



Visual Development Tools

CASE Tools (Cont.)



A class diagram from IBM's Rational Rose

105

CASE Tools

- Categories of such CASE products referred to as
  - Reverse Engineering
    - Refers to the process of creating design specifications for a system or program module from program code and data definitions.
  - Reengineering tools
    - They are similar to reverse engineering tools but include analysis features that can automatically, or interactively with System Analyst, alter an existing system in an effort to improve its quality or performance

CASE Tools

SDLC Phase	Key Activities	CASE Tool Usage
Project identification and selection	Display and structure high-level organizational information	Diagramming and matrix tools to create and structure information
Project initiation and planning	Develop project scope and feasibility	Repository and documentation generators to develop project plans
Analysis	Determine and structure system requirements	Diagramming to create process, logic, and data models
Logical and physical design	Create new system designs	Form and report generators to prototype designs; analysis and documentation generators to define specifications
Implementation	Translate designs into an information system	Code generators and analysis, form and report generators to develop system; documentation generators to develop system and user documentation
Maintenance	Evolve information system	All tools are used (repeat life cycle)

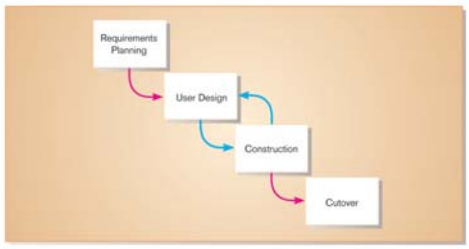
107

Rapid Application Development (RAD)

- RAD is based on building prototypes that evolve into finished systems (often using time boxing)
  - A prototype is a smaller-scale, representative or working model of the users' requirements or a proposed design for an information system.
  - A time box is a non-extendable period of time, usually 60-120 days, by which a candidate system must be placed into operation.
- Methodology to radically decrease design and implementation time.

108

Rapid Application Development (RAD)



109

Rapid Application Development (RAD)

- RAD claims to offer
  - a development lifecycle for much faster systems development
  - better and cheaper systems
  - more rapid deployment of systems as developers and users work together in real time

Rapid Application Development (RAD)

- RAD relies on
  - extensive user involvement
  - JAD sessions
  - Prototyping
  - I-CASE tools (integrated CASE tools)
  - Code generators

Rapid Application Development (RAD)

- Evolution of RAD
  - Pressures for businesses to speed up and compete in a changing, global environment
  - Shorter development lifecycles
  - Dissatisfaction with IT department
  - Diffusion of high-powered prototyping and CASE tools
    - Why wait 2 or 3 years to develop systems likely to be obsolete upon completion?

Rapid Application Development (RAD)

- James Martin's four pillars of RAD
  - Tools
  - People
  - Methodology
  - Management

Rapid Application Development (RAD)

- Tools
  - I-CASE tools with prototyping and code generation facilities
  - Visual development environments
- People
  - Manager and user participation in JAD type workshops
  - Developer roles - workshop leader, project leader, scribe, repository manager, construction or SWAT (Skilled With Advanced Tools) team

Rapid Application Development (RAD)

- Methodology
  - to guide and control the use of RAD techniques
  - Should be automated for ease of use - adaptability and flexibility
- Management
  - Executive sponsor
  - Facilities and support for the RAD team

Rapid Application Development (RAD)

- RAD lifecycle
  - Is evolutionary
  - Uses “timeboxing”
  - Avoids “feature creep”
  - Avoids requirements “gold plating”

Rapid Application Development (RAD)

- RAD lifecycle
  - Requirements planning phase (JRP)
  - User design phase (JAD)
  - Construction phase
  - Cutover phase

Rapid Application Development (RAD)

- Martin’s (1991) RAD lifecycle
  - Requirements planning phase
    - managers, executives, key users determine requirements in terms of business areas and business problems
    - JRP workshops to agree requirements, overall planning

Rapid Application Development (RAD)

- Martin’s (1991) RAD lifecycle
  - User design phase
    - end users and IS personnel use I-CASE for rapid prototyping of system design
    - JAD sessions to develop basis for physical design
    - users sign off on CASE-based design (no paper-based spec)

Rapid Application Development (RAD)

- Martin’s (1991) RAD lifecycle
  - Construction phase
    - IS personnel now generate code using I-CASE tool
    - end users validate screens, design, etc.
  - Cutover phase
    - delivery of new system to users: testing, training, implementation
    - can be combined with construction in small systems

Rapid Application Development (RAD)

- Uses timebox approach:
  - system to be developed divided into components that can be developed separately
  - the easiest and most important 75% of the system functionality produced in first timebox (90 day cycle)
  - forces users to focus on the necessary and most well-defined aspects
  - users experience this component first and other component requirements may then change
  - functionality is trimmed: “gold plating” is avoided
  - avoids “feature creep” - more and more requirements creep in during development than originally specified

Rapid Application Development (RAD)

- Timeboxing vs traditional approach
  - Traditional approach - every possible requirement is implemented together leading to increased complexity and long delays
  - Martin claims RAD can produce a system in 6 months that would take 24 months using traditional development methods
  - Small development teams are essential for RAD to work

Rapid Application Development (RAD)

- Advantages
  - quick development
  - cost savings
  - higher quality/improved performance as easier and most important functions targeted first
  - avoids feature creep
  - aligned with business changes

Rapid Application Development (RAD)

- Disadvantages
  - detailed business models/understanding neglected --> inconsistencies, misunderstandings
  - programming standards, scalability, system administration issues neglected e.g. database maintenance, database reorganisation, backup/recovery, distribution of system updates, etc

Application Packages

- Purchasing or leasing set of pre-written application software programs that are commercially available
- May range from simple PC systems to complex mainframe or client-server systems

Application Packages

- Useful
  - when you need an information system for a common company function eg. payroll
  - when information systems resources for in-house development are in short supply
  - when the application software package is more cost effective than in-house development
  - because the most of the design and implementation tasks are done - significant time saving
  - because the system and documentation are usually maintained by the vendor

Application Packages

- Useful
  - because the design spec is fixed - no endless reworking - users have to accept it politically because:
    - external work is often perceived as being superior to an in-house effort - easier to get new systems into the company
    - easier to get management support because of fixed costs
    - problems can be attributed to the package rather than internal sources - ends endless source of internal conflict

Application Packages

- Limitations
  - very rare to find a package that can do everything well that a user wants
  - often need to develop specialised package additions because multi-purpose packages do not handle certain functions well
  - conversion and integration costs can sometimes be so significant as to render the project infeasible

Application Packages

- Limitations
  - some vendors refuse to support packages which have been customised by the users - and most packages require some customisation
  - customisation can be so extensive that it would have been cheaper to develop the system in-house

Commercial Off-the-Shelf Software Route

- Commercial off-the-shelf (COTS) software is a software package or solution that is purchased to support one or more business functions and information systems.

Agile Methodologies

- Three key principles
  - Focus on adaptive methodologies (not predictive)
  - Focus on people (not roles)
  - Focus on self-adaptive processes (refine and improve development process)

131

Agile Methodologies

- Argues that engineering and software development are not the same.
  - Engineering requirements are well understood.
  - Software requirements are not well understood and change continually.

132

Agile Methodologies (Cont.)

TABLE 1-1 The Agile Manifesto

The Manifesto for Agile Software Development

Seventeen anarchists agree:

We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:

- Individuals and interactions over processes and tools.
- Working software over comprehensive documentation.
- Customer collaboration over contract negotiation.
- Responding to change over following a plan.

That is, while we value the items on the right, we value the items on the left more.

We follow the following principles:

- Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
- Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.
- Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
- Business people and developers work together daily throughout the project.
- Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.
- The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.
- Working software is the primary measure of progress.
- Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.
- Continuous attention to technical excellence and good design enhances agility.
- Simplicity—the art of maximizing the amount of work not done—is essential.
- The best architectures, requirements, and designs emerge from self-organizing teams.
- At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

—Kent Beck, Mike Beedle, Arie van Bennekum, Alistair Cockburn, Ward Cunningham, Martin Fowler, James Grenning, Jim Highsmith, Andrew Hunt, Ron Jeffries, Jon Kern, Brian Marick, Robert C. Martin, Steve Mellor, Ken Schwaber, Jeff Sutherland, Dave Thomas (www.agileAlliance.org)

(Source: From Fowler and Highsmith, 2001. Used by permission.)

133

eXtreme Programming

- Short, incremental development cycles.
- Automated tests.
- Two-person programming teams.

134

eXtreme Programming (Cont.)

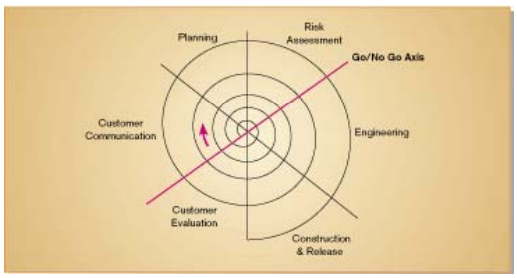
- Coding and testing operate together.
- Advantages:
  - Communication between developers.
  - High level of productivity.
  - High-quality code.

135

Spiral Approach

- The circular nature of the life cycle diagram illustrates how the end of the useful life of one system leads to the beginning of another project that will replace the existing system altogether.
- This concept is clearly brought out with the diagram shown next. Each of these phase has specific outcomes and deliverables that feed important information to other phases. At the end of each phase, the systems development reaches a mile stone.

Spiral Approach



Object-Oriented Analysis and Design (OOAD)

- Based on objects rather than data or processes.
- Object: a structure encapsulating attributes and behaviors of a real-world entity.

138

Object-Oriented Analysis and Design (OOAD) (Cont.)

- Object class: a logical grouping of objects sharing the same attributes and behaviors.
- Inheritance: hierarchical arrangement of classes enable subclasses to inherit properties of superclasses.

139

Rational Unified Process (RUP)

- An object-oriented systems development methodology.
- RUP establishes four phase of development: inception, elaboration, construction, and transition.
- Each phase is organized into a number of separate iterations.

140

Phases of OOSAD-based Development

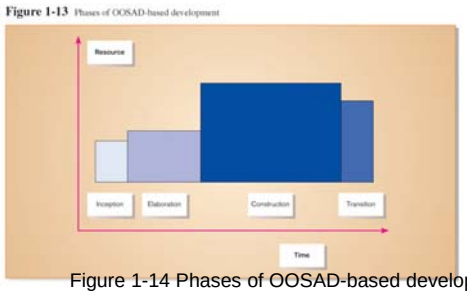


Figure 1-14 Phases of OOSAD-based development

141

Our Approach to Systems Development

- The SDLC is an organizing and guiding principle in this book.
- We may construct artificial boundaries or artificially separate activities and processes for learning purposes.
- Our intent is to help you understand all the pieces and how to assemble them.

142

Summary

- In this chapter you learned how to:
  - Define information systems analysis and design.
  - Describe the different types of information systems.
  - Describe the information Systems Development Life Cycle (SDLC).
  - Explain Rapid Application Development (RAD), prototyping, Joint Application Development (JAD), and Computer Aided Software Engineering (CASE).
  - Describe agile methodologies and eXtreme programming.
  - Explain Object Oriented Analysis and Design and the Rational Unified Process (RUP).

143