# Chapter 1
# Industrial Information Systems - Introduction (part II)
# مقدمة لأنظمة المعلومات الصناعية (جزء ثان)

King Saud University
College of Engineering
IE department

Dr. Eng. Shady Aly

# Information system development
# عملية بناء نظام المعلومات

# Introduction to IS development

- Systems development methodology is a standard process followed in an organization to analyze, design, implement and maintain information systems

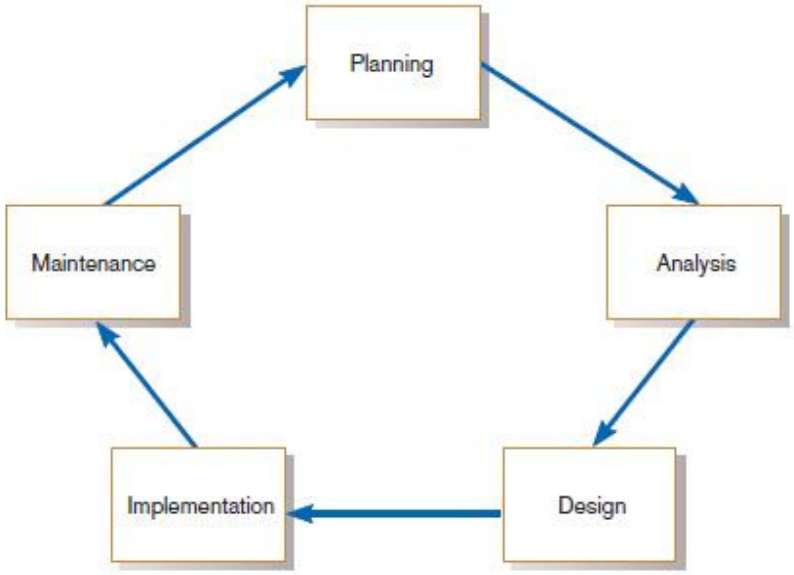
- System analyst is responsible for analysis and design of information systems

# System development life cycle (SDLC)
# دورة حياة بناء وتطوير النظام

- A traditional methodology used to plan, analyze, design, implement and maintain information systems

# SDLC

- Phases in SDLC:
  - Planning
  - Analysis
  - Design
  - Implementation
  - Maintenance



# SDLC- Cont.

- **Planning** – an organization's total information system objectives or purpose are identified, analyzed, prioritized, and arranged

- **Analysis** – system requirements are studied and structured

# SDLC- Cont.

- **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
  - **Physical design** : transforming the logical specifications of the system into the technology-specific details

# SDLC- Cont.

- **Implementation** – the information system is coded, tested, installed and supported in the organization

- **Maintenance** – an information system is systematically repaired and improved

# Types of SDLCs

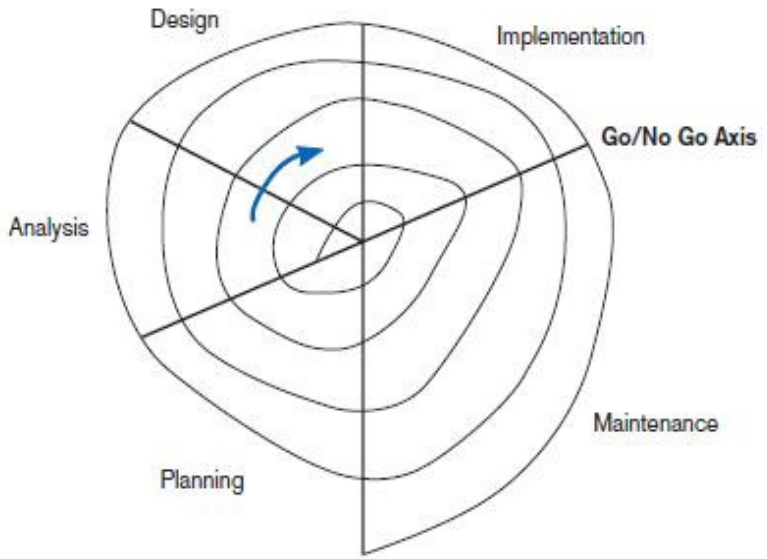
- SDLC can be performed in two ways:
  - Iterative SDLC (على محاولات – مراحل تحسينية)
  - Traditional Waterfall SDLC

# Iterative SDLC

- Development phases are repeated as required

until an acceptable system is found

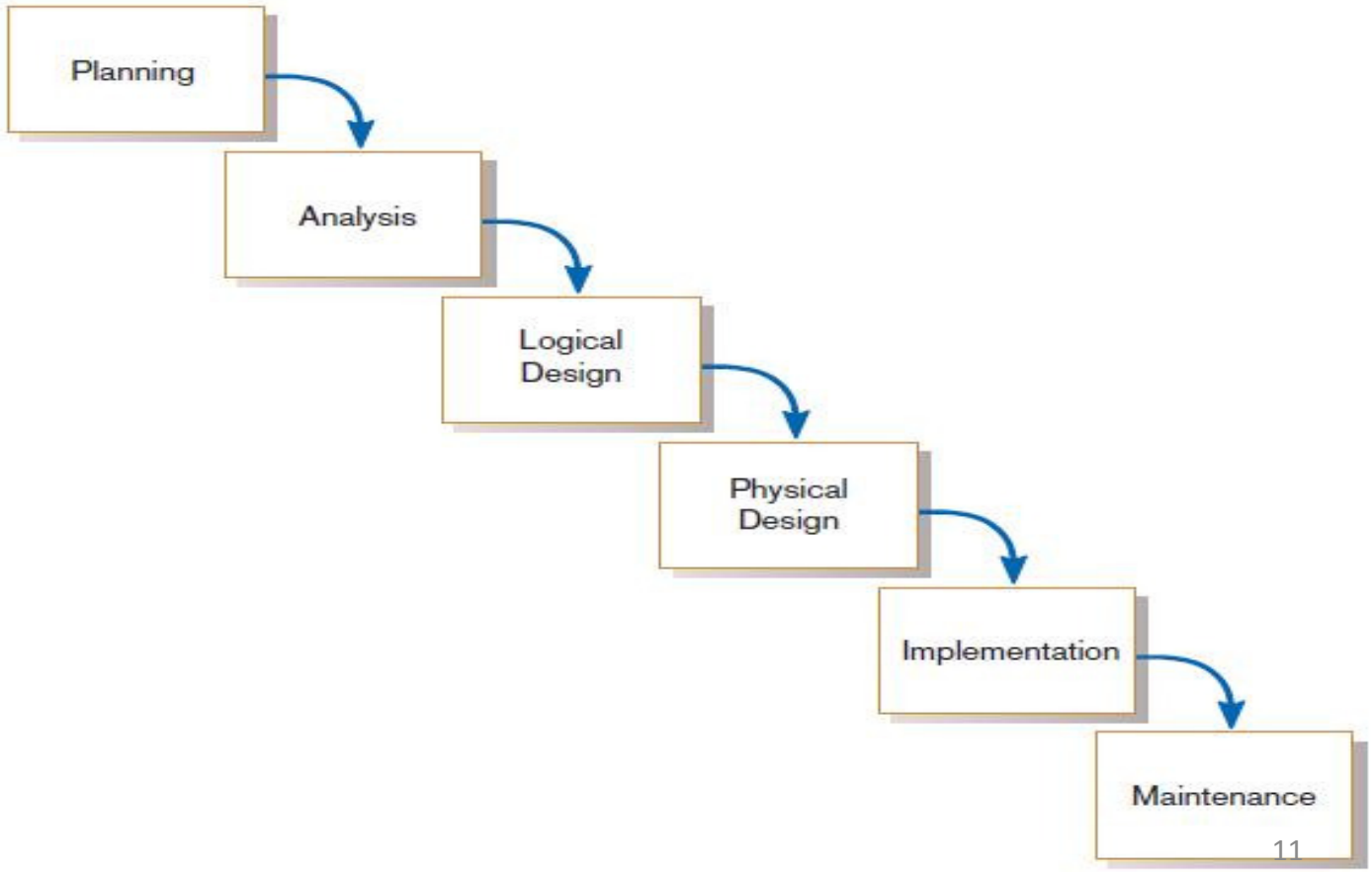
- User participates




- Spiral development حلزوني(evolutionary) SDLC in which we constantly cycle through the phases at different levels of details

# Traditional Waterfall SDLC

- One phase begins when another completes, with little backtracking (رجوع للخلف) and looping.



# Problems with Waterfall Approach

- System requirements after being determined can't change
- Limited user cooperation (only in requirements phase)

# Different Approaches to Improving IS Development

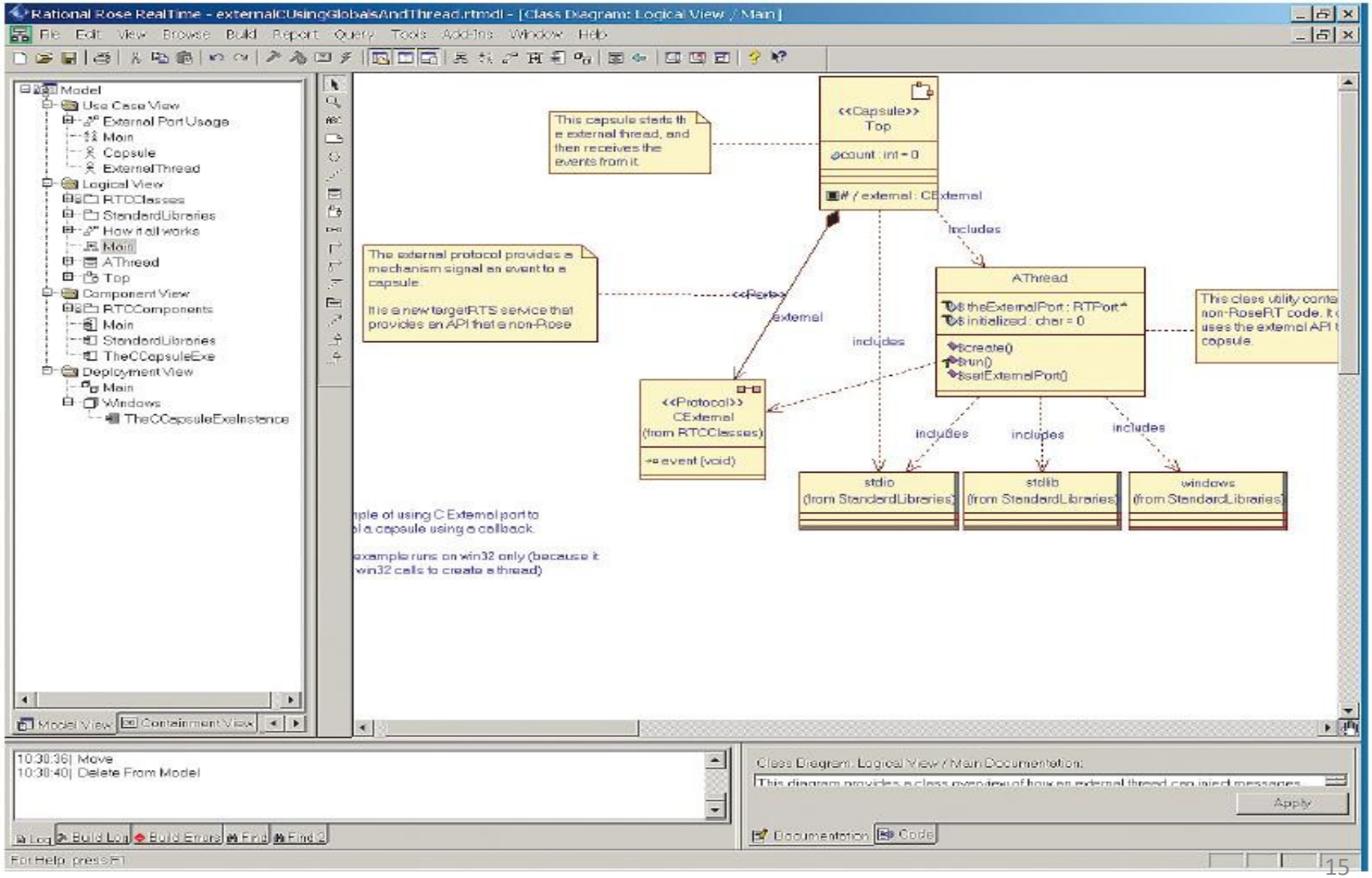
- CASE (Computer-aided Software Engineering) Tools
  - بناء البرمجيات بمساعدة الحاسب
- Rapid Application Development (RAD)

  التطوير السريع للنظام

# Computer-Aided Software Engineering (CASE) Tools

- Diagramming tools enable graphical representation.
- Computer displays and report generators help prototype how systems “look and feel”.
- Documentation generators (توثيق التعليمات الفنية و المستخدم) standardize technical and user documentation.
- Code generators (البرمجة) enable automatic generation of programs and database code directly from design documents, diagrams, forms, and reports.

# Computer-Aided Software Engineering (CASE) Tools

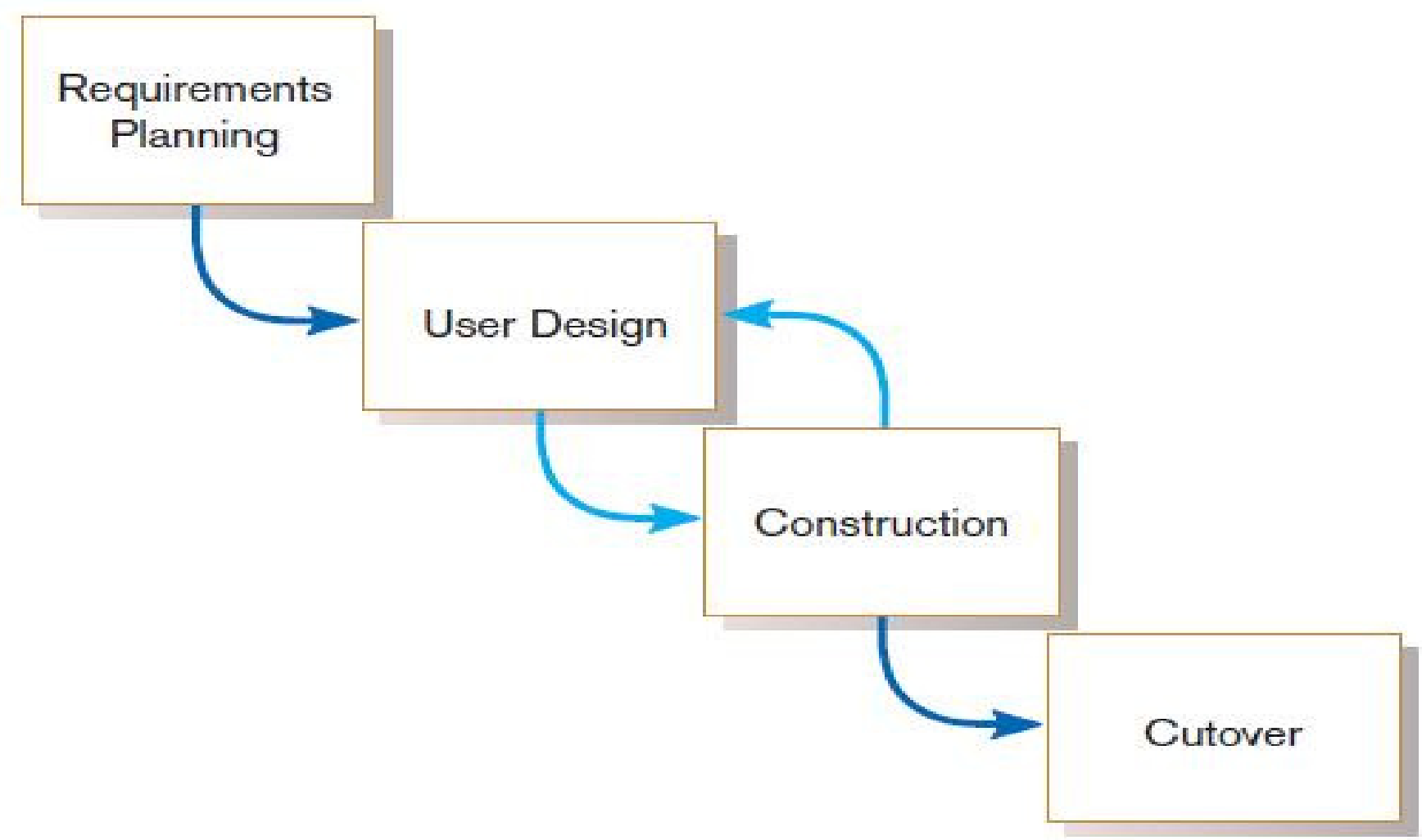


# Rapid Application Development (RAD)

- Methodology to radically decrease design and implementation time…shortened development

- It has extensive user cooperation, prototyping, integrated CASE tools, and code generators

# Rapid Application Development (RAD)



# Rapid Application Development (RAD)

- **Requirements planning**: planning the requirements with user focusing on the functions and systems interfaces and reports

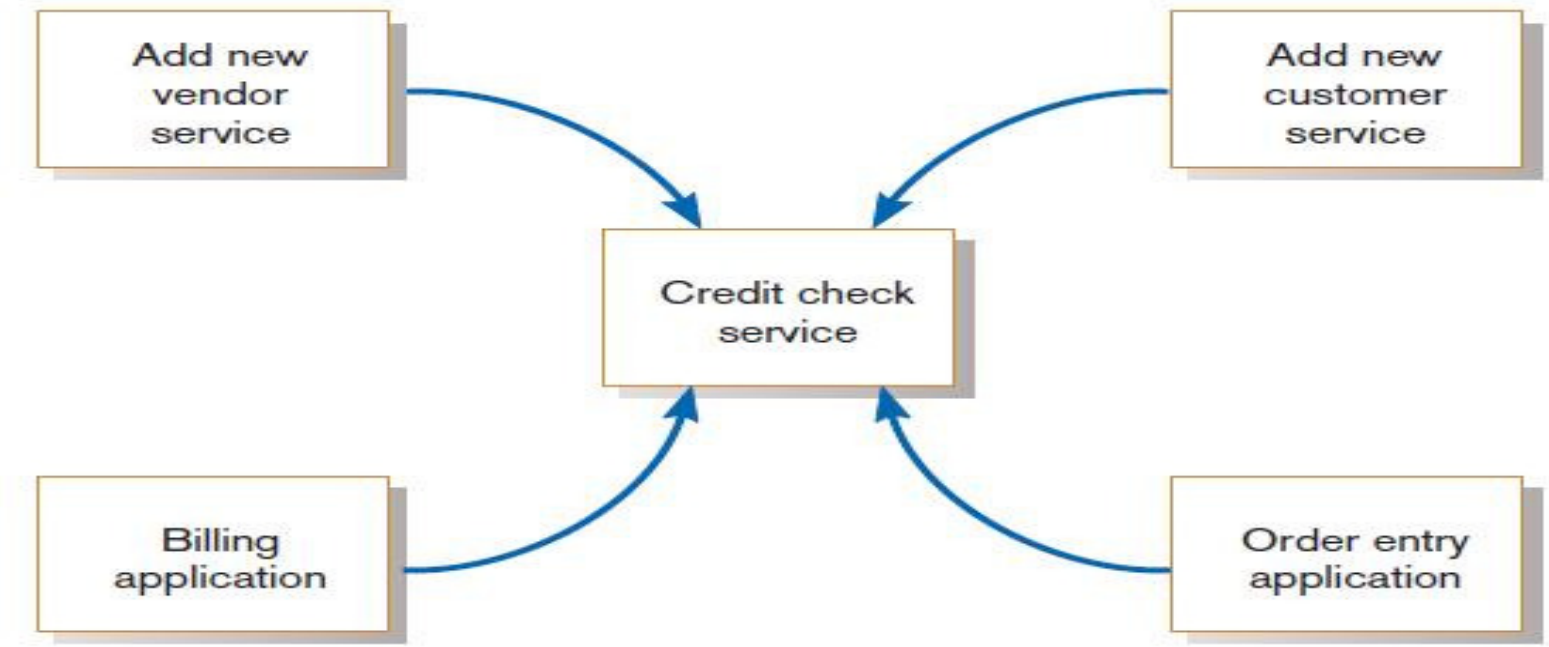
- **User design**: prototyping the system with user using CASE in creating interfaces and report

- **Construction**: coding the system using CASE

- **Cutover**: delivery of the developed system to its end user

# Service-Oriented Architecture (SOA)

- A method to systems development based on building complete systems through assembling software components , each of which model generic business functions:



# Object-Oriented Analysis and Design (OOAD) النمذجة الشيئية أو نمذجة الأهداف

- Based on objects rather than data or processes
  - **Object**: a structure encapsulating attributes and behaviors of a real-world entity.
  - **Object class**: a logical grouping of objects sharing the same attributes and behaviors
  - **Inheritance**: hierarchical arrangement of classes enable subclasses to inherit properties of super-classes

# The sources of software
# مصادر البرمجيات

# Sources of software

- There are various sources of software for organizations.

- There are criteria to evaluate software from different sources.

# Sources of Software

- Information technology services firm (**Outsourcing**)
- Packaged software producers
- Enterprise-wide solutions
- Cloud Computing
- Open source software
- In-house development

# Information Technology (IT) Services Firms

- **Outsourcing** (الإستقدام أو الجلب من الخارج): Turning over responsibility نقل مسئولية of some or all of an organization's information systems applications and operations to an outside firm

- **Reasons to outsource**
  - Cost-effective
  - Take advantage of economies of scale
  - Free up internal resources
  - Reduce time to market
  - Increase process efficiencies
  - When system development is a non-core activity for the organization

- Help companies develop custom information systems for internal use.

- Develop, host, and run applications for customers.

- Provide other services.

# Packaged / off-the-shelf Software Producers

- Serve many market segments (متنوع الإستخدام).
- Provide software ranging from broad-based packages to specialized (متخصص) packages.
- Software runs on all size computers, from microcomputers to large mainframes.
- Prepackaged software (e,g., MS. Project) is off-the-shelf, turnkey (جاهز) software (i.e. not customizable غير قابل للتعديل).

# Enterprise Solutions Software

- **Enterprise Resource Planning (ERP)** systems integrate individual traditional business functions into modules enabling a single seamless transaction to cut across functional boundaries.

- SAP AG is the leading vendor of ERP systems.

# Cloud Computing

- The provision of computing resources, including applications, **over the Internet**, so customers do not have to invest in the computing infrastructure needed to run and maintain the resources

# Open Source Software

- Freely available including source code
- Developed by a community of interested people (making money through maintenance, support and selling fully-featured versions)
- Performs the same functions as commercial software
- Examples: Linux, mySQL, Firefox

# In-House Development

## التطوير الذاتي داخل المؤسسة

- If sufficient system development expertise with the chosen platform exists in-house, then some or all of the system can be developed by the organization's own staff.

- Hybrid solutions involving some purchased and some in-house components are common.

# Comparing sources of software components

Comparison of Six Different Sources of Software Components

| Producers | When to Go to This Type of Organization for Software | Internal Staffing Requirements |
|---|---|---|
| IT services firms | When task requires custom support and system can't be built internally or system needs to be sourced | Internal staff may be needed, depending on application |
| Packaged software producers | When supported task is generic | Some IS and user staff to define requirements and evaluate packages |
| Enterprise-wide solutions vendors | For complete systems that cross functional boundaries | Some internal staff necessary but mostly need consultants |
| Cloud computing | For instant access to an application; when supported task is generic | Few; frees up staff for other IT work |
| Open source software | When supported task is generic but cost is an issue | Some IS and user staff to define requirements and evaluate packages |
| In-house developers | When resources and staff are available and system must be built from scratch | Internal staff necessary though staff size may vary |

# Criteria Selecting Off-the-Shelf (جاهز) Software

- **Cost**: comparing the cost of developing in-house with the cost of purchasing or licensing the software pack
- **Functionality**: the tasks that the software can perform
- **Vendor support**: how much support the vendor provide and at what cost
- **Viability of vendor (بقاء المورد)**: can the software adapt to changes in systems software and hardware
- **Flexibility**: how easy it is to customize the software
- **Documentation**: is the user's manual and technical documentation understandable and up-to-date
- **Response time**: how long it takes the software package to respond to the user's requests in an interactive session
- **Ease of installation**: a measure of the difficulty of loading the software and making it operational