LECTURE 1. SYSTEMS DEVELOPMENT

1.1 INFORMATION SYSTEMS

System
- A system is an interrelated set of business procedures used within one business unit working together for a purpose
- A system has nine characteristics
  - A system exists within an environment
  - A boundary separates a system from its environment

Characteristics of a System
- Components
- Interrelated Components
- Boundary
- Purpose
- Environment
- Interfaces
- Constraints
- Input
- Output

Information Technology
- Combination of computer technology (hardware and software) with telecommunications technology (data, image, and voice networks)

Information Systems
- Turns data into information
- Includes:
  - Hardware and system software
  - Documentation and training materials
  - Job roles associated with the system
  - Controls to prevent theft or fraud
  - The people who use the software to perform their jobs

1.2 SYSTEMS DEVELOPMENT LIFE CYCLE

Major Attributes of the Lifecycle
- The project
  - Moves systematically through phases where each phase has a standard set of outputs
  - Produces project deliverables
  - Uses deliverables in implementation
  - Results in actual information system
  - Uses gradual refinement

Project Phases
- Planning
  - Why build the system? How should the team go about building it?
  - Identifying business value
  - Analyze feasibility
  - Develop work plan
  - Staff the project
  - Control and direct project

- Analysis
- Who uses system, what will it do, where and when will the system be used?
- Information gathering
- Process modeling
- Logic modeling
- Data modeling

- **Design**
  - How will the system work?
  - Physical design
  - Architectural design
  - Interface design
  - Database and file design
  - Program design

- **Implementation**
  - System delivery
  - Construction
    - Program building
    - Program and system testing
  - Installations
    - Conversion strategy
    - Training plan
    - Support plan
  - Operation
    - System changed to reflect changing conditions
    - System obsolescence

**Systems Development Life Cycle**
- Phases are not necessarily sequential
  - Sequential
  - Parallel
- Each phase has a specific outcome and deliverable
- Individual companies use customized life cycle

**Processes and Deliverables**
- Planning
  - System Request
  - Feasibility Analysis
  - Workplan
- Analysis → System Proposal
- Design → System Specification
- Implementation → New System and Maintenance Plan

1.3 IMPLEMENTING SDLC

**Methodology**
- A formalized approach to implementing the SDLC
  - A series of steps and deliverables
- Methodology Categories

<table>
<thead>
<tr>
<th>Category I</th>
<th>Category II</th>
</tr>
</thead>
<tbody>
<tr>
<td>Process Oriented</td>
<td>Structured Development</td>
</tr>
<tr>
<td>Data Oriented</td>
<td>Rapid Application Development</td>
</tr>
<tr>
<td>Object Oriented</td>
<td>Agile Development</td>
</tr>
</tbody>
</table>

**Waterfall Development Methodology**
- Structured
• Proceed in sequence from one phase to another
  • Pros
    • Identifies systems requirements long before programming begins
    • Minimizes changes to requirements as project progresses
  • Cons
    • Design must be specified on paper before programming begins
    • Long time between system proposal and delivery of new system

Parallel Development Methodology
  • General design
  • Divide project into subprojects that are designed and implemented in parallel
  • Final integration
  • Pros
    • Reduces Schedule Time
    • Less Chance of rework
  • Cons
    • Still uses paper documents
    • Sub-projects May Be Difficult to Integrate

Rapid Application Development
  • Incorporate special techniques and tools:
    • Joint Application Design (JAD)
      - Users, Managers and Analysts work together for several days
      - System requirements are reviewed
      - Structured meetings
    • CASE tools
      - Automate or support drawing and analysis of system models
      - Translate of system models into application programs
    • CASE repository: system developers’ database for system models, detailed descriptions and specifications, and other products of system development
    • Forward engineering: draw system models that are subsequently transformed into program code.
    • Reverse engineering: read existing program code and transform that code into a representative system model that can be edited and refined by the systems analyst.
      • Fourth generation/visualization programming languages
      • Code generators

Three RAD Categories
  • Phased development
    • A series of versions developed sequentially
  • Prototyping
    • System prototyping
  • Throw-away prototyping
    • Design prototyping

Phased Development Methodology
  • Break into a series of versions that are developed sequentially
  • Pros
    • User get a system to use quickly
    • User can identify additional needs for later versions
  • Cons
    • Users work with a system that is intentionally incomplete
Prototyping
- Building a scaled-down working version of the system
- Concurrent analysis, design and implementation, repeated
- Advantages:
  - Users are involved in design
  - Captures requirements in concrete form
- Pros
  - Users interact with prototype very quickly
  - Users can identify needed changes and refine real requirements
- Cons
  - Tendency to do superficial analysis
  - Initial design decisions may be poor

Throwaway Prototyping
- Design prototype
- Pros
  - Risks are minimized
  - Important issues are understood before the real system is built
- Cons
  - May take longer than prototyping

Agile Development: Extreme Programming
- Programming centric
- Streamline SDLC by eliminating modeling and documentation overheads
- XP: coding and continuous testing by pairs of developers
- Iterative, system functionalities grow over time
- Pros
  - Fast delivery of results
  - Works well in projects with undefined or changing requirements
- Cons
  - Requires discipline
  - Works best in small projects
  - Requires much user input

Criteria for Selecting the Appropriate Methodology
- Clear user requirements
- Familiarity with technology
- Complexity of system
- Reliability of system
- Time schedule
- Schedule visibility

1.4 TEAM ROLES AND SKILLS

Role of Systems Analyst
- Study problems and needs of an organization
- Determine best approach to improving organization through use of:
  - People
  - Methods
  - Information technology
- Help system users and managers define their requirements for new or enhanced systems
- Assess options for system implementation
  - In-house development
  - Outsourced development
  - Outsourced development and operation
  - Commercial application
- For in-house projects, work on a team of analysts and developers
Skills of a Successful Systems Analyst

○ Analytical
  ○ Understanding of organizations
  ○ General business knowledge
  ○ Problem solving skills
  ○ System thinking
    ▪ Ability to see organizations and information systems as systems

○ Technical
  ○ Understanding of potential and limitations of technology
  ○ Working knowledge of information technology
  ○ Computer programming experience and expertise
  ○ Systems analysis and design skills

○ Managerial
  ○ Ability to manage projects, resources, risk and change

○ Interpersonal
  ○ Effective written and oral communication skills
  ○ Interpersonal relations skills
  ○ Flexibility and adaptability
  ○ Character and ethics