Lesson 1: System Integration

What is System Integration?

- The process of combining different subsystems or components into a unified whole.
- In technology and software development, it involves making software systems and hardware devices work together cohesively.

Key Aspects of System Integration

1. **Connectivity** – Ensures seamless data flow between subsystems.
2. **Data Flow** – Involves creating communication pathways between different systems.
3. **Functionality** – The integrated system should perform functions that individual components cannot achieve alone.
4. **Interoperability** – Ensures components from different vendors or developers work together despite differing standards.
5. **Testing** – Rigorous testing identifies and resolves integration issues.
6. **Scalability** – Integrated systems should be scalable for future growth without disruptions.

Integrated Program Planning

- Specific enterprise systems integrate data across functions, supporting organizational operations.

Enterprise Resource Planning (ERP) System

- Integration is a cornerstone in today's enterprise environments with multiple ERP systems.

Evolution of ERP

- **1960s**: Inventory Management and Control
- **1970s**: Material Requirements Planning (MRP)
- **1980s**: Manufacturing Requirements Planning (MRP II)
- **1990s**: Enterprise Resource Planning (ERP)
- **2000s**: Extended ERP (ERP II)

Five Key Questions in the Planning Stage

1. What data does the target system require for integration?
2. Where is the data located, and what transformations are needed?
3. What constitutes a transaction, and what dependencies exist?
4. How will you connect to the target system, and what security constraints apply?
5. What interface options are available for integration?

Can be simply to…

Data Requirements – Understanding data needed for the integration task is fundamental.

Data Mapping and Transformations – Identify data location in the source system and the transformations needed for compatibility.

Transaction Definition and Dependencies – Defining transactions and understanding dependencies helps ensure data integrity during integration.

Connection and Security – Establish secure connections and manage credentials to protect data integrity.

Interface Options – Choose from available interface options to enable system communication (e.g., REST, SOAP).

Lesson 2: Program Execution

Logical System Integration

- Allows organizations to share data with stakeholders based on need and authorization.

Physical System Integration

- Provides seamless connectivity between heterogeneous systems.

Middleware

- Software that provides a seamless data presentation to the user while maintaining data integrity and synchronization.

Program Integration

- Addresses the market demand and ensures effective system integration.

Steps in Integration System

1. **Resource Categorization** – Take an inventory of hardware and software resources, including vendors and platforms.
2. **Compliance and Standards** – Ensure support for standards such as JDBC/ODBC for databases.
3. **Legacy Systems Support** – Develop policies to support older systems.
4. **Middleware Tools** – Utilize middleware for integration when older systems must be maintained temporarily.
5. **Authentication and Authorization Policies** – Develop a single sign-on policy for integrated system access.
6. **Centralized IT Services and Help Desk Support** – Provide centralized IT support for seamless operations.
7. **Backup, Recovery, and Security Policies** – Plan for data recovery in the event of system failure.
8. **Hardware and Software Standardization Policies** – Establish policies for hardware and software acquisition.

Benefits of System Integration

- Increased revenue, growth, and competitive advantage.
- Enhanced information visibility and standardization.

Limitations of System Integration

- High initial setup costs.
- Power and interdepartmental conflicts.
- Long-term, intangible ROI.
- Limitation in creativity.

Lesson 3: Program Management

Program Management Information

- The Project Management Office (PMO) ensures project teams are synchronized and addressing functionality issues efficiently.

Critical Success Factors

1. **Decision-Making Process** – A clear decision-making process minimizes scope, efficiency, and productivity issues.
2. **Project Scope** – Well-defined project scope prevents scope creep and ensures the project meets its goals.
3. **Teamwork** – Collaboration across teams is essential for success.
4. **Change Management** – Effective communication and training are critical for managing change.

Managing Scope Creep

- "Change Control" involves managing changes through a formal process and governance. Options, costs, and timeframes should be documented for any scope changes.

Implications for Management

- The success or failure of a project often depends on the PMO's ability to manage scope, ensure good communication, and maintain team continuity.

Lesson 4: Generic Program Preparation

Continuous Process Improvement

- ERP implementation success depends on redesigning processes rather than customizing technology to fit them.

System Development Life Cycle (SDLC)

1. **Planning**
2. **Analysis**
3. **Design**
4. **Implementation**
5. **Testing & Integration**
6. **Maintenance**

Traditional SDLC

- A structured process that involves identifying problems, designing solutions, and implementing systems in a top-down approach.

ERP Implementation Life Cycle

- ERP applications are prepackaged software developed to automate and integrate business processes. It differs from personal software by being tailored for organizational needs.

ERP Implementation Plan Choices

1. **Comprehensive** – Full ERP functionality with major business process reengineering (BPR) and customization.
2. **Middle of the Road** – Balanced approach with moderate BPR.
3. **Vanilla** – Minimal customization, relying on core ERP functionality and best practices.

ERP Life Cycle vs SDLC

- ERP emphasizes customizing software and changing business processes rather than determining user requirements as in the SDLC.

Traditional ERP Life Cycle Stages

1. **Scope and Commitment Stage** – Develop project scope, plan, and management commitment.
2. **Analysis and Design Stage** – Analyze user requirements, conduct gap analysis, and design changes.
3. **Acquisition and Development Stage** – Configure the platform, execute gap analysis tasks, and customize software.
4. **Implementation Stage** – Install and release the system to users, with a focus on training and system conversion.
5. **Operation Stage** – Provide ongoing support and updates, manage new releases, and monitor user feedback.