System Integration & Architecture

1

Introduction

- Many systems are built to easy, improve and transform organizations.
- Some organizations have many departments which run systems which are independent of each other.
- And systems built sometimes, may not have an abstract view (architecture) which leads to failure of system interoperability.
- There is need to have architectural view of the system as a priority to help in the design to avoid the likeliness of system failure.

2

Introduction

- Besides after the system has been designed and developed in consideration of the size of the organization, i.e. most especially when the organization is large, need is required to integrate such systems to ensure flexibility, Speed, Cost , Standardization, Data integrity, reliability and robustness.
- This can help Information Technology (IT), energy, and financial services industry among others to have an easy to use integrated system.

3

What students need to know

- Systems Integration (SI) process, approaches, drivers, tools and techniques required for successful SI, critical success factors, and best practices.
- The course focuses on how a proposed system will be integrated with other existing or planned systems.
- It addresses the System Integration problem using architectures as the basis and then addresses the evaluation of the architectures in terms of the capabilities they provide.

4

What students need to learn

- The theory and practice of business process integration, legacy integration, new systems integration, business-to-business integration, integration of commercial-off-the-shelf (COTS) products, interface control and management, testing, integrated program management, integrated Business Continuity Planning (BCP).

5

Aims

- To provide the students an understanding of the technical and business process issues involved in systems integration.

6

Learning outcomes

- On completion of this course, the students will be able to:
  - Identify integration issues upfront in the process of System Integration and should be able to identify the best practices that ensure successful System Integration.
  - Have an understanding of the technical and business process issues involved in systems integration.

7

Teaching and learning pattern

- Teaching this course will be in lecture form. A number of case studies will also be used to illustrate some concepts as mentioned in the indicative content.

8

Indicative content

- The System of Systems Integration Problem
  - Human, Organizational, Societal Cultural, Economic, and Technological aspects;
    - Processes, approaches, drivers, tools and techniques required for successful SI, critical success factors, and best practices in Systems Integration;
    - The Role of Architectures in Systems Integration;
    - Integration in a System of Systems and a Federation 60 of Systems;
    - Model Based Architecture, Design, and Integration;
    - Systems of Systems Interoperability;
    - Evaluation of architectures;
    - Measures of Performance and Effectiveness;

9

Indicative content

- Assessment of System Capabilities;
- Analysis of Alternatives;
- Case studies and examples from the Information Technology (IT), energy, and financial services industry to illustrate the concepts discussed.

- The theory and practice of business process integration, legacy integration, new systems integration, business-to-business integration, integration of commercial-off-the-shelf (COTS) products, interface control and management, testing, integrated program management, integrated Business Continuity Planning (BCP). Specific focus will be given to issues of interface integration and interoperability of systems.

10

Assessment method

- Assessment will be in form of tests and practical assignments (40%) and final written examination (60%)

11

Reference books

- David D. Walden, Garry J. Roedler, Kevin J. Forsberg, Thomas M. Shortell, Systems engineering handbook : a guide for system life cycle processes and activities, 4th Edition, WILEY Publishing, 2015..

12

Key terminologies in this course

- Various key terminologies shall be used throughout this course as follows
- System
- Systems thinking
- System Integration
- System Architecture
- Project

13

System

- An array of components designed to accomplish a particular objective according to plan. Many sub-systems many be designed which later on are combined together to form a system which is intended to achieve a specific objective which may be set by the Project manager.

14

Systems thinking

- Is a way of understanding an entity in terms of its purpose, as three steps
- The three major steps followed in systems thinking

1. Identify a containing whole (system), of which the thing to be explained is a part.
2. Explain the behavior or properties of the containing whole.
3. Explain the behavior or properties of the thing to be explained in terms of its *role(s)or function(s) within its containing whole*

(Ackoff, 1981)

15

System Integration

- Is the combination of inter-related elements to achieve a common objective (s).

16

System Architecture

- The architecture of a system defines its high-level structure, exposing its gross organization as a collection of interacting components.
- Elements needed to model a software architecture include:
  - Components, Connectors, Systems, Properties and Styles.

17

What is a project?

- From the key terms described above, a system developer and architects cannot do anything without first establishing various projects. These projects may be new or existing.
- So it is inevitable to first understand what a project is, factors that influence the project, who the owners are and many more as discussed below.

18

What Is a Project?

- A project is a temporary endeavor undertaken to accomplish a unique product or service
- Attributes of projects
  - unique purpose
  - temporary
  - require resources, often from various areas
  - should have a primary sponsor and/or customer
  - involve uncertainty

19

Where do information Systems Projects Originate (Sources of Projects)?

New or changed IS development projects come from **problems, opportunities**, and **directives** and are always subject to one or more ***constraints***.

1. **Problems** – may either be current, suspected, or anticipated. Problems are undesirable situations that prevent the business from fully achieving its purpose, goals, and objectives (users discovering real problems with existing IS).
2. An **Opportunity** – is a chance to improve the business even in the absence of specific problems. This means that the business is hoping to create a system that will help it with increasing its revenue, profit, or services, or decreasing its costs.
3. **A Directive** – is a new requirement that is imposed by management, government, or some external influence i.e. are mandates that come from either an internal or external source of the business.

20

Projects Cannot Be Run in Isolation

- Projects must operate in a broad organizational environment
- Project managers need to take a holistic or systems view of a project and understand how it is situated within the larger organization

Stakeholders

- Stakeholders are the people involved in or affected by project activities
- Stakeholders include
  - the project sponsor and project team
  - support staff
  - customers
  - users
  - suppliers
  - opponents to the project

Importance of Stakeholders

- Project managers must take time to identify, understand, and manage relationships with all project stakeholders
- Using the four frames of organizations can help meet stakeholder needs and expectations
- Senior executives are very important stakeholders

23

Table 2-2. What Helps Projects Succeed?

According to the Standish Group's report "CHAOS 2001: A Recipe for Success," the following items help IT projects succeed, in order of importance:

- **Executive support**
- User involvement
- Experienced project manager
- Clear business objectives
- Minimized scope
- Standard software infrastructure
- Firm basic requirements
- Formal methodology
- Reliable estimates

24

Understanding Organizations

We can analyze a formal organization using the following 4 (fou frames;

Structural frame:	**Human resources frame:**
Focuses on roles and responsibilities, coordination and control. Organizational charts help define this frame.	Focuses on providing harmony between needs of the organization and needs of people.
Political frame:	**Symbolic frame:**
Assumes organizations are coalitions composed of varied individuals and interest groups. Conflict and power are key issues.	Focuses on symbols and meanings related to events. Culture is important.

25

Many Organizations Focus on the Structural Frame

- Most people understand what organizational charts are
- Many new managers try to change organizational structure when other changes are needed
- 3 basic organizational structures
  - Functional-
  - project
  - matrix

26

Basic Organizational Structures

- Organizational structure depends on the company and/or the project.
- The structure helps define the roles and responsibilities of the members of the department, work group, or organization.
- It is generally a system of tasks and reporting policies in place to give members of the group a direction when completing projects.
- A good organizational structure will allow people and groups to work effectively together while developing hard work ethics and attitudes.
- The four general types of organizational structure are functional, divisional, matrix and project-based.

27

Basic Organizational Structures

- **Functional Structure** - People who do similar tasks, have similar skills and/or jobs in an organization are grouped into a functional structure. The advantages of this kind of structure include quick decision making because the group members are able to communicate easily with each other. People in functional structures can learn from each other easier because they already possess similar skill sets and interests.
- **Divisional Structure -** In a divisional structure, the company will coordinate inter-group relationships to create a work team that can readily meet the needs of a certain customer or group of customers. The division of labor in this kind of structure will ensure greater output of varieties of similar products. An example of a divisional structure is geographical, where divisions are set up in regions to work with each other to produce similar products that meet the needs of the individual regions.

28

Basic Organizational Structures

- **Matrix Structure** - Matrix structures are more complex in that they group people in two different ways: by the function they perform and by the product team they are working with. In a matrix structure the team members are given more autonomy and expected to take more responsibility for their work. This increases the productivity of the team, fosters greater innovation and creativity, and allows managers to cooperatively solve decision-making problems through group interaction.
- **Project Organization Structure -** In a project-organizational structure, the teams are put together based on the number of members needed to produce the product or complete the project. The number of significantly different kinds of tasks are taken into account when structuring a project in this manner, assuring that the right members are chosen to participate in the project.

29

Basic Organizational Structure

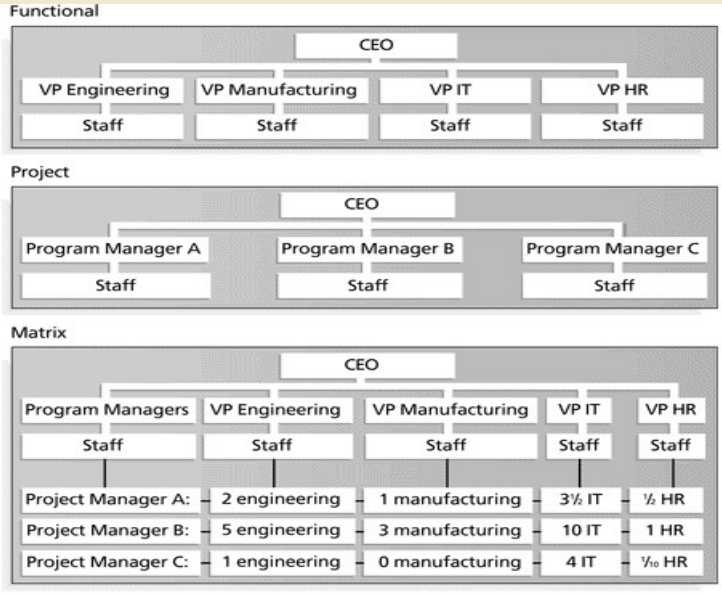


Figure 2-2. Functional, Project, and Matrix Organizational Structures

30

Project Phases and the Project Life Cycle

- A project life cycle is a collection of project phases
- Project phases vary by project or industry, but some general phases include
  - concept
  - development
  - implementation
  - support

Phases of the Project Life Cycle

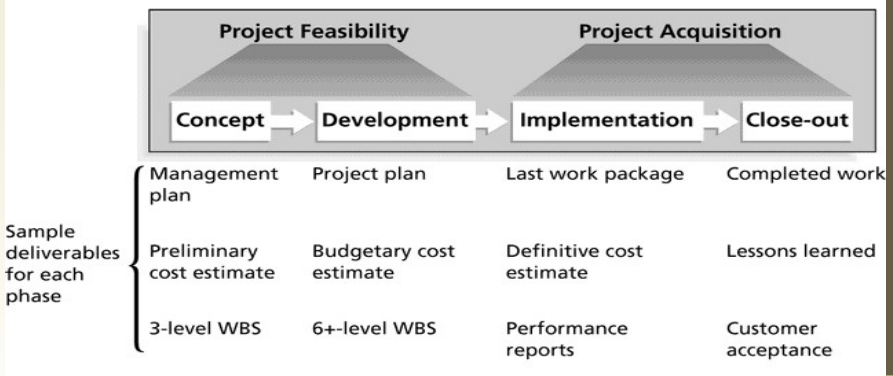


Figure 2-3. Phases of the Project Life Cycle

Product Life Cycles

- Products also have life cycles
- **The Systems Development Life Cycle (SDLC)** is a framework for describing the phases involved in developing and maintaining information systems
- **Systems development projects can follow**
  - **Predictive models:** The scope of the project can be clearly articulated and the schedule and cost can be predicted.
  - **Adaptive models:** Projects are mission driven and component based, using time-based cycles to meet target dates.

33

Predictive Life Cycle Models

- The waterfall model has well-defined, linear stages of systems development and support.
- The spiral model shows that software is developed using an iterative or spiral approach rather than a linear approach.
- The incremental release model provides for progressive development of operational software.
- The prototyping model is used for developing prototypes to clarify user requirements.
- The RAD model is used to produce systems quickly without sacrificing quality.

34

Adaptive Life Cycle Models

- **Extreme Programming (XP):** Developers program in pairs and must write the tests for their own code. XP teams include developers, managers, and users.
- **Scrum:** Repetitions of iterative development are referred to as sprints, which normally last thirty days. Teams often meet every day for a short meeting, called a scrum, to decide what to accomplish that day. Works best for object-oriented technology projects and requires strong leadership to coordinate the work

35

Distinguishing Project Life Cycles and Product Life Cycles

- The project life cycle applies to all projects, regardless of the products being produced
- Product life cycle models vary considerably based on the nature of the product
- Most large IT systems are developed as a series of projects
- Project management is done in all of the product life cycle phases

36

Why Have Project Phases and Management Reviews?

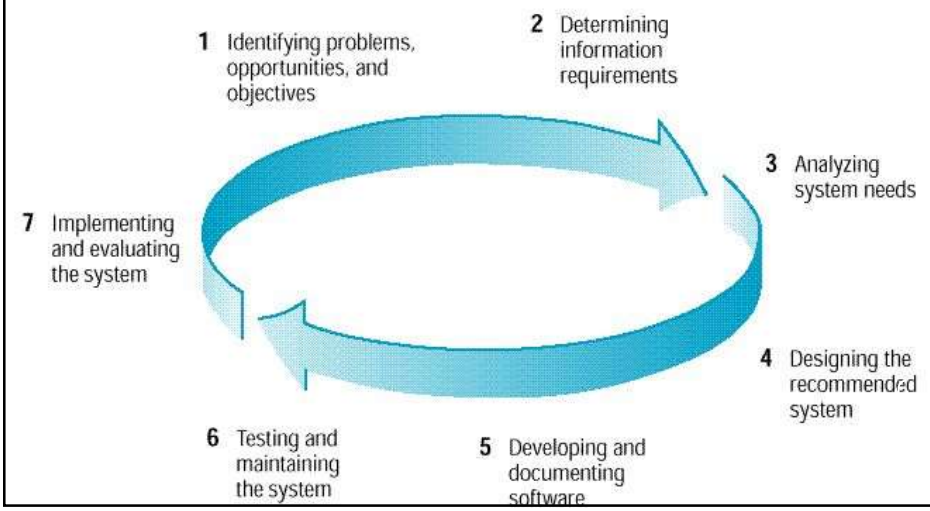
- A project should successfully pass through each of the project phases in order to continue on to the next
- Management reviews (also called phase exits or kill points) should occur after each phase to **evaluate the project's progress**, **likely success**, and **continued compatibility** with organizational goals

37

System Development Life Cycle

(Kendall & Kendall terminology)

Figure 1.2 The Seven Phases of the Systems Development Life Cycle



38

User Requirements (User Needs)

39

What are requirements?

- Requirements are statements that identify the essential needs of a system in order for it to have value and utility.

40

Requirements

- A system cannot be analyzed, designed, implemented and evaluated unless the problem is understood and requirements elicited.
- Requirements are fundamental basis of all the system development processes.
- System architects will always base of the requirements elicited by the system analyst to design an architectural view of the system.
- There is need some integrations: business process integration, legacy integration, new systems integration, business-to-business integration, integration of commercial-off-the-shelf (COTS) products, interface control and management, testing, integrated program management, integrated Business Continuity Planning (BCP), requirement is the basis.

41

Characteristics of Good Req'ts

- 1. Describes What, Not How.
- 2. Atomic. i.e., it should have a single purpose
- 3. Unique.
- 4. Documented and Accessible.
- 5. Identifies Its Owner.
- 6. Approved. After a requirement has been revised, reviewed, and rewritten, it must be approved by its owner.
- 7. Traceable. A good requirement is traceable; it should be possible to trace each requirement back to its source.
- 8. Necessary.

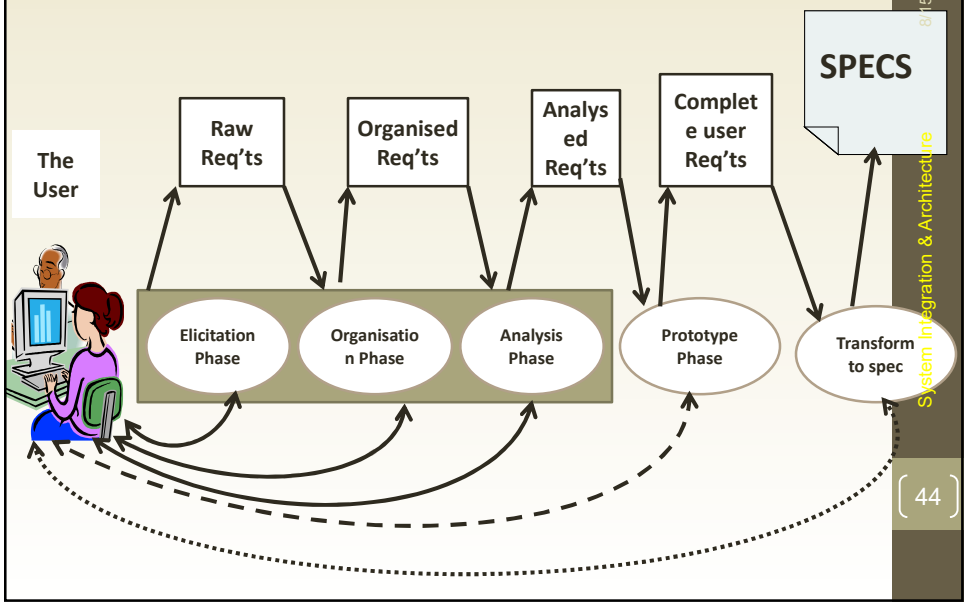
42

Characteristics of Good Req'ts cont.

- 9. Complete.
- 10. Unambiguous
- 11. Quantitative and testable
- 12. Identifies applicable states
- 14. States Assumptions. All assumptions should be stated.
- 15. Use of Shall, Should, and Will. A mandatory requirement should be expressed using the word shall (e.g., "The system shall conform to all state laws
- 16. **Avoids Certain Words**. The words optimize, maximize, and minimize should not be used in stating requirements, because we could never prove that we had achieved them.

43

Requirements Life cycle



44

Requirement Life Cycle .. Cont..

- **Elicitation Phase**

The starting point of the requirements engineering process is an elicitation process that involves a number of people to ensure consideration of a broad scope of potential ideas and candidate problems

- **Organisation Phase**

In this step there is no transformation of the requirements, but simp classification and categorization. For example, requirements may b grouped into functional vs. nonfunctional requirements.

- **Analysis Phase**

This represents a transformation.

45

Requirement Life Cycle .. Cont..

- **Prototype Phase**

In this way poorly understood requirements may be tested and perhaps strengthened, corrected, or refined.

- **Requirements documentation and specification**

This represents the requirements as the finished product of the stakeholder requirements team.

The requirements are compiled into a requirements list or into some equivalent document format. These collected requirements are then transformed into a specification.

46

Requirements elicitation, documentation, and maintenance

47

Requirements elicitation

- Requirements determination **addresses the gathering and documenting** of the true and real requirements for the Information System being developed.
- Requirements is the wants and /or needs of the user within a problem domain. elicit

48

Requirements determination questions

- Requirements determination questions
  - Who does it?
  - What is done?
  - Where is it done?
  - When is it done
  - How is it done
  - Why is it done?

49

Systems Requirements

- Characteristics or features that must be included to satisfy business requirements
  - Outputs
  - Inputs
  - Processes
  - Timing
  - Controls
  - Volumes. sizes, and frequencies
- Data/Information collected can be about; people, organisation, work and work environment.

50

Fact – Finding Methods

- Sampling (of existing documentation, forms, and databases).
- Research and site visits. (Participation)
- Observation of the work environment.
- Questionnaires.
- Interviews.
- Prototyping.
- JAD/Joint requirements planning (JRP).

51

Types of Requirements

- **User Requirements:** these are statements in Natural language plus diagrams of services the system provides, together with its operational constraints. These can be categorised into 2; functional requirements and non-functional requirements
  - **Functional requirements**
    - Describe ***what*** the system should do
  - **Non-functional requirements**
    - Consists of ***Constraints*** that must be adhered to during development (design and implementation)
    - Remember '**Constraints**.'
- **System requirements**
  - What we agree to provide
  - Describes system services
  - Contract between Client and contractor

52

Functional requirements

- What *inputs* the system should accept
- What *outputs* the system should produce
- What data the system should *store* that other systems might use
- What *computations* the system should perform
- The *timing and synchronization* of the above

53

Non-functional requirements

- Non-functional requirements are global constraints on a computer system
  - e.g. development costs, operational costs, performance, reliability,
- The challenge of Non-functional requirements:
  - Hard to model
  - Usually stated informally, and so are:
    - often contradictory,
    - difficult to enforce during development
    - difficult to evaluate for the customer prior to delivery

54

Non-functional requirements

- Define system properties and constraints e.g. reliability, response time and storage requirements. Constraints are I/O device capability, system representations.
- Process requirements may also be specified mandating a particular programming language or development method
- Non-functional requirements may be more critical than functional requirements. If these are not met, the system is useless.

55

Examples of NFR

- Interface requirements
  - how will the new system interface with its environment?
  - User interfaces and "user-friendliness"
  - Interfaces with other systems
- Performance requirements
  - Time - response time
  - Throughput - transactions per second

56

Examples of NFR

- Security
  - permissible information flows
  - Or who can do what
  - Survivability – e.g. system will need to survive fire natural catastrophes, etc
- Operating requirements
  - physical constraints (size, weight),
  - personnel availability & skill level
  - accessibility for maintenance
  - environmental conditions

57

Examples of NFR

- Lifecycle requirements
  - Maintainability, Enhanciability, Portability, expected market or product lifespan
- limits on development
  - E.g. development time limitations, resource availability and methodological standards.
- Economic requirements
  - e.g. restrictions on immediate and/or long-term costs.

58

Requirements Documentation

- There are basically two types of documents realised from the requirements elicitation phase. These include;
  - User Requirements Specification Document
  - System requirements specification Document

59

User Requirements Specification – URS/URD

- The URS document outlines precisely what the User (or customer) is expecting from this system.
- User Requirement Specification may incorporate the functional requirements of the system or may be in a separate document labelled the Functional Requirements Specification - the FRS.

The URD has the following information:

1. Functional Requirements
2. Non-Functional Requirements

60

System Requirements Specification Document

A detailed description of the system services.

- What do we agree to provide?
- A structured document setting out detailed descriptions of the system services.
- Written as a contract between client and contractor.

61

TOOLS THAT AID IN DEVELOPING & UNDERSTANDING SYSTEM REQ'TS

- Affinity diagrams
- Force-field analysis
- Ishikawa fishbone (cause-and-effect) diagrams
- Pareto diagrams
- Pugh charts
- Quality function deployment (QFD)

62

Template of User Needs

Funtional Requirements

Stakeholders	Descriptions	Ideas	Priority
1. Admin	Manage Web site	OK	1
1.1 magage the accounts	Each use of website has one account in this system	OK	1
1.2 Reset password	User need reset password to inform addmin by email, SMS	OK	1

Non Funtional Requirements

Stakeholders	Descriptions	Ideas	Priority
1. Admin	Manage Web site	OK	1
1.1 Password policies	Password must be complcated	OK	1
1.2 User name policies	User name: First Letter in Capital letter; Unit	OK	1

63