System Engg.

What is a system?

Definition: System is a purposeful collection of inter-related components working together towards some common objective.

OR

It is an interrelated set of components, with an identifiable boundary, working together for some purpose.

- ✓ A system may include software, mechanical, electrical and electronic hardware and be operated by people.
- ✓ System components are dependent on other system components
- ✓ Every system interacts with the environment through exchange of information.
- ✓ Every system must fit into its operating environment.
- ✓ All systems exhibit predictable behavior.
- ✓ All systems come in hierarchies. Therefore, system exists to maintain its higher level systems.

Characteristics of a System:

Components: An irreducible part or aggregation of parts that make up a system, also called subsystem

Interrelated Components: Dependence of one subsystem on one or more subsystems.

Boundary: The line that marks the inside and outside of a system and that sets off the system from its environment.

Purpose: The overall goal or function of a system.

Environment: Everything external/internal to a system that interacts with the system.

Interfaces: Point of contact where a system meets its environment or where subsystems meet each other

Input: Whatever a system takes from its environment in order to fulfill its purpose.

Output: Whatever a system returns to its environment in order to fulfill its purpose.

Constraints: A limit to what a system can accomplish

Note:

- ✓ ***Open system:*** *A system that interacts freely with its environment, taking input and returning output.*
- ✓ ***Closed System:*** *A system that is cut off from its environment and does not interact with* ***it***

A computer-based system makes use of a variety of system elements which includes:

Software: Computer programs, data structures, and related documentation that serve to effect the logical method, procedure, or control that is required.

Hardware: Electronic devices that provide computing capability, the interconnectivity devices (e.g., network switches, telecommunications devices) that enable the flow of data, and electromechanical devices (e.g., sensors, motors, pumps) that provide external world function.

People: Users and operators of hardware and software.

Database: A large, organized collection of information that is accessed via software.

Documentation: Descriptive information (e.g., hardcopy manuals, on-line help files, Web sites) that portrays the use and/or operation of the system.

Procedures: The steps that define the specific use of each system element or the procedural context in which the system resides.

Emergent system properties:

- ✓ Properties of the system as a whole rather than properties that can be derived from the properties of components of a system

- ✓ Emergent properties are a consequence of the relationships between system components

- ✓ They can therefore only be assessed and measured once the components have been integrated into a system

Types of emergent property

- **Functional properties**

  - ✓ These appear when all the parts of a system work together to achieve some objective. For example, a bicycle has the functional property of being a transportation device once it has been assembled from its components.

- **Non-functional emergent properties**

- ✓ Examples are reliability, performance, safety, and security. These relate to the behaviour of the system in its operational environment. They are often critical for computer-based systems as failure to achieve some minimal defined level in these properties may make the system unusable

Examples of emergent properties

- **The volume of the system**
  - The volume (total space occupied) varies depending on how the component assemblies are arranged and connected.
- **The reliability of the system**
  - This depends on the reliability of system components and the relationships between the components but unexpected interactions can cause new types of failure and affect the reliability of the system.
- **The Security of the system**
  - The ability to resist attack means security. It is a complex property that cannot be easily measured.
- **The Reparability of the system**
  - This property reflects how easy it is to fix a problem with the system once it has been discovered. It depends on being able to diagnose the problem, access components that are faulty and modify or replace these components.

✓ ***The usability of a system***

- This property reflects how easy it is to use the system which is not simply dependent on the system hardware and software but also depends on the system operators and the environment where it is used.

Influences on reliability

✓ ***Hardware reliability***

- What is the probability of a hardware component failing and how long does it take to repair that component?

✓ ***Software reliability***

- How likely is it that a software component will produce an incorrect output? Software failure is usually distinct from hardware failure in that software does not wear out.

✓ ***Operator reliability***

- How likely is it that the operator of a system will make an error?

Reliability relationships

✓ Hardware failure can generate spurious signals that are outside the range of inputs expected by the software

✓ Software errors can cause alarms to be activated which cause operator stress and lead to operator errors

✓ The environment in which a system is installed can affect its reliability

The ‘shall-not’ properties

- Properties such as performance and reliability can be measured
- However, some properties are properties that the system should not exhibit
  - Safety - the system should not behave in an unsafe way
  - Security - the system should not permit unauthorised use
- Measuring or assessing these properties is very hard

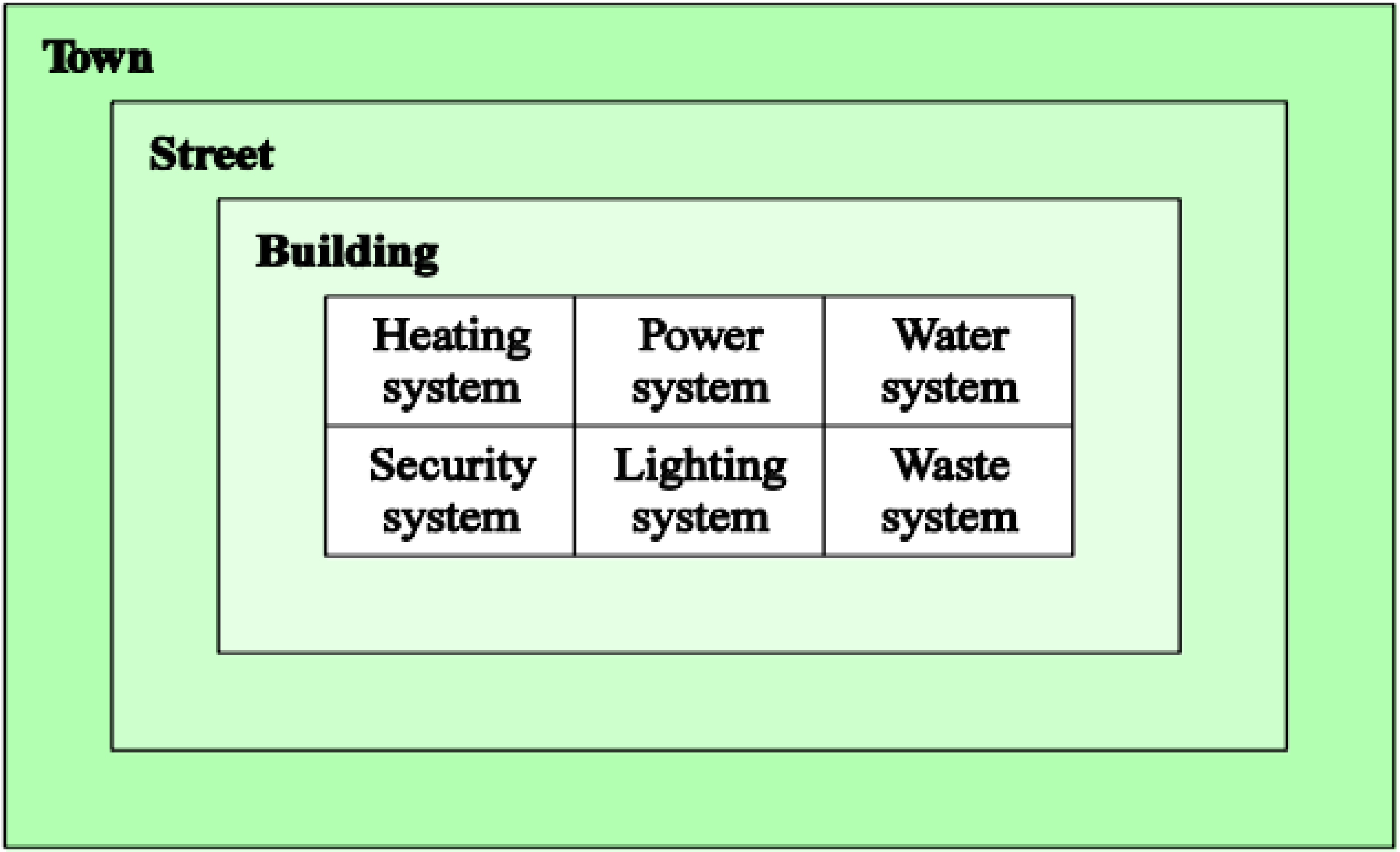
- Properties such as performance and reliability can be measured
- However, some properties are properties that the system should not exhibit
  - Safety - the system should not behave in an unsafe way
  - Security - the system should not permit unauthorised use
- Measuring or assessing these properties is very hard

Systems and their environment

- ✓ Systems are not independent but exist in an environment
- ✓ System’s function may be to change its environment
- ✓ Environment affects the functioning of the system e.g. system may require electrical supply from its environment
- ✓ The organizational as well as the physical environment may be important

System hierarchies

System hierarchies



Town

Street

Building

Heating system	Power system	Water system
Security system	Lighting system	Waste system

Human and organizational factors in Organizations

Process changes

- Does the system require changes to the work processes in the environment?

Job changes

- Does the system de-skill the users in an environment or cause them to change the way they work?

Organisational changes

Does the system change the political power structure in an organisation?

Systems Engineering:

Systems engineering is the activity of specifying, designing, implementing, validating, deploying and maintaining socio-technical systems.

Discipline involved in System engineering:

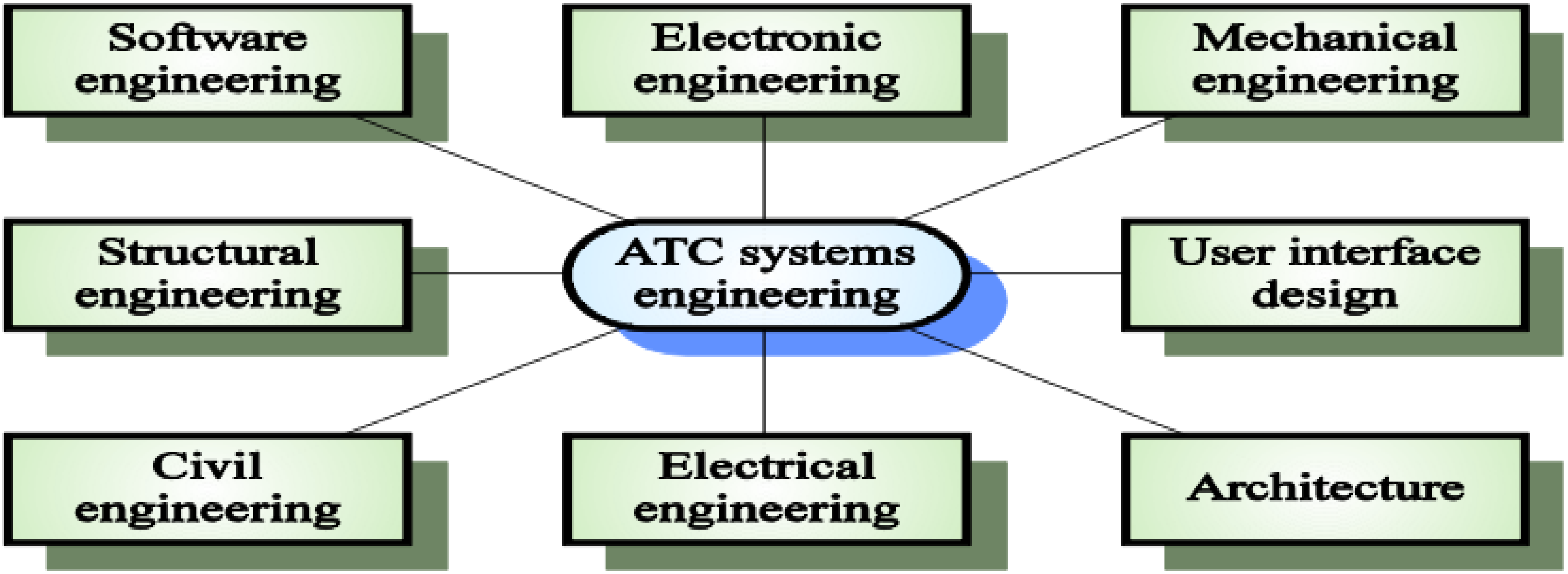


Fig: Discipline involved in System Engineering (Air Traffic Control)

Software and systems engineering

- ✓ The proportion of software in systems is increasing. Software-driven general purpose electronics is replacing special-purpose systems
- ✓ Problems of systems engineering are similar to problems of software engineering
- ✓ Software is (unfortunately) seen as a problem in systems engineering. Many large system projects have been delayed because of software problems

Problems of systems engineering

- ✓ Large systems are usually designed to solve 'wicked' problems
- ✓ Systems engineering requires a great deal of co-ordination across disciplines

- Almost infinite possibilities for design trade-offs across components
- Mutual distrust and lack of understanding across engineering disciplines

Systems must be designed to last many years in a changing environment

Missed schedule

Improper integration of subsystems

Maintenance problems

Unmanageable systems.

The system engineering process

- ✓ A set of activities whose goal is the development or evolution of software
- ✓ Usually follows a 'waterfall' model because of the need for parallel development of different parts of the system
  - o Little scope for iteration between phases because hardware changes are very expensive. Software may have to compensate for hardware problems
- ✓ Inevitably involves engineers from different disciplines who must work together
  - o Much scope for misunderstanding here. Different disciplines use a different vocabulary and much negotiation is required. Engineers may have personal agendas to fulfil

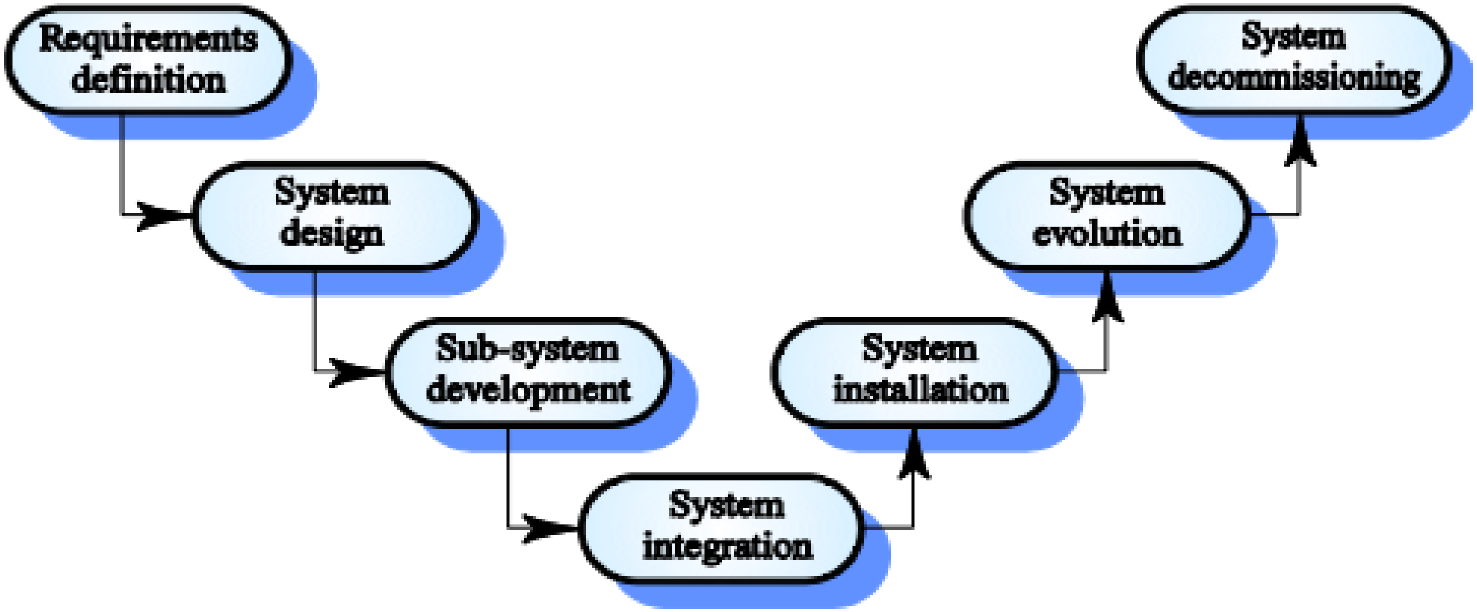


Fig: Phases of the system engineering process

1. System requirements definition

System requirements definitions specify what the system should do (its functions) and its essential and desirable system properties. Creating system requirements definitions involves consultations with system customers and end-users.

This phase concentrates on three types of requirements:

✓ **Abstract functional requirements**: The basic function that the system must provide are defined at an abstract level. More detailed functional requirements specification takes place at the sub-system level.

✓ **System properties**: These are non-functional emergent system properties such as availability, performance and safety. These properties affect the requirements for all sub-systems.

✓ **Characteristics that the system must not exhibit:** It is sometimes as important to specify what the system must not do as it is to specify what the system should do.

An important part of the requirements definition phase is to establish a set of overall objectives that the system should meet. These should not necessarily be expressed in terms of the system's functionality but should define why the system is being procured for a particular environment.

System requirements problems:

- ✓ Changing as the system is being specified
- ✓ Must anticipate hardware/communications developments over the lifetime of the system
- ✓ Hard to define non-functional requirements (particularly) without an impression of component structure of the system

2. The system design process

System design is concerned with how the system functionality is to be provided by the components of the systems. The activities involved in this process are:

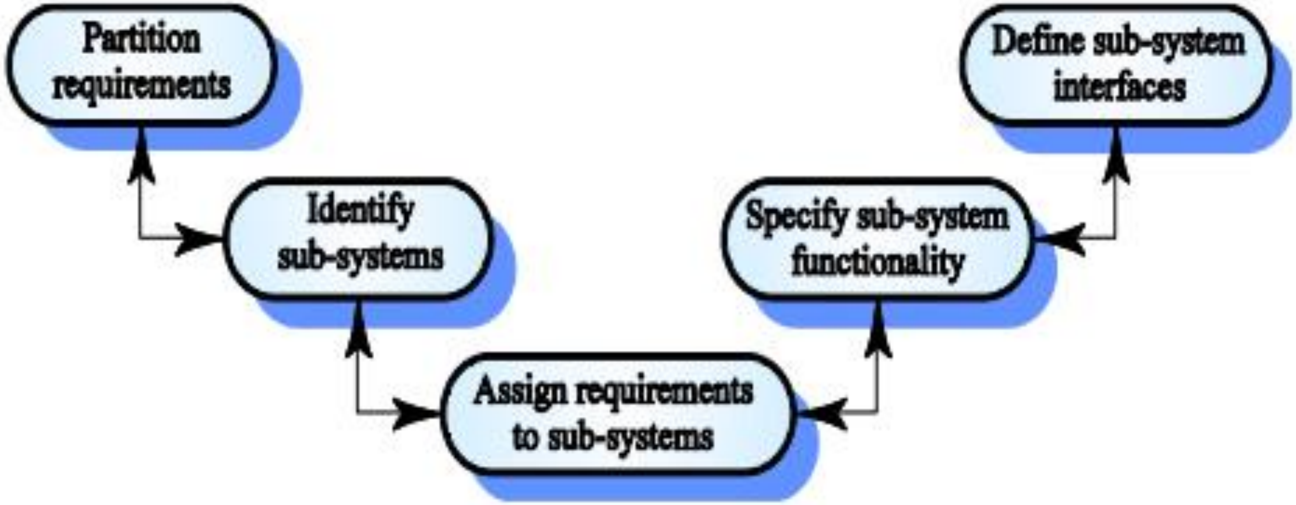

Partition requirements
Identify sub-systems
Assign requirements to sub-systems
Specify sub-system functionality
Define sub-system interfaces

Fig: The system design process

- ✓ **Partition requirements**
  - Organise requirements into related groups
- ✓ **Identify sub-systems**
  - Identify a set of sub-systems which collectively can meet the system requirements

- **Assign requirements to sub-systems**
  - Causes particular problems when COTS (Commercial Off-the-Shelf)are integrated
- **Specify sub-system functionality**
  - Specify the specific function provided by each sub-system and also try to identify relationships between subsystems.
- **Define sub-system interfaces**
  - Critical activity for parallel sub-system development

System design problems

- ✓ Requirements partitioning to hardware, software and human components may involve a lot of negotiation

- ✓ Difficult design problems are often assumed to be readily solved using software

- ✓ Hardware platforms may be inappropriate for software requirements so software must compensate for this

3. Sub-system development

✓ Typically, parallel projects developing the hardware, software and communications

✓ May involve some COTS (Commercial Off-the-Shelf) systems procurement

✓ Lack of communication across implementation teams

✓ Bureaucratic and slow mechanism for proposing system changes means that the development schedule may be extended because of the need for rework

4. System integration

✓ It is the process of putting hardware, software and people together to make a system

✓ It should be tackled incrementally so that sub-systems are integrated one at a time

✓ Interface problems between sub-systems are usually found at this stage

✓ May be problems with uncoordinated deliveries of system components

5. System installation

- ✓ The organizational process of changing over from the current system to a new one

Four approaches of installation

- **Direct Installation**

  Changing over from the old system to a new one by turning off the old system when the new one is turned on

- **Parallel Installation**

  Running the old system and the new one at the same time until management decides the old system can be turned off

- **Single location installation**

  Trying out a system at one site and using the experience to decide if and how the new system should be deployed throughout the organization

- **Phased Installation**

  Changing from the old system to the new one incrementally, starting with one or a few functional components and then gradually extending the installation to cover the whole new system

Problems in installation:

- ✓ Environmental assumptions may be incorrect
- ✓ May be human resistance to the introduction of a new system
- ✓ System may have to coexist with alternative systems for some time
- ✓ May be physical installation problems (e.g. cabling problems)
- ✓ Operator training has to be identified

6. System evolution

- ✓ Large, complex systems have a very long lifetime. They must evolve to meet changing requirements i.e. during their life, they are changed to correct errors in the original system requirements and to implement new requirements that have emerged.
- ✓ System Evolution is inherently costly, like software evolution for several reasons:
  - *Changes must be analysed from a technical and business perspective*
  - *Sub-systems interact so unanticipated problems can arise*

- *There is rarely a rationale for original design decisions*
- *System structure is corrupted as changes are made to it*

✓ Existing systems which must be maintained are sometimes called **legacy systems**

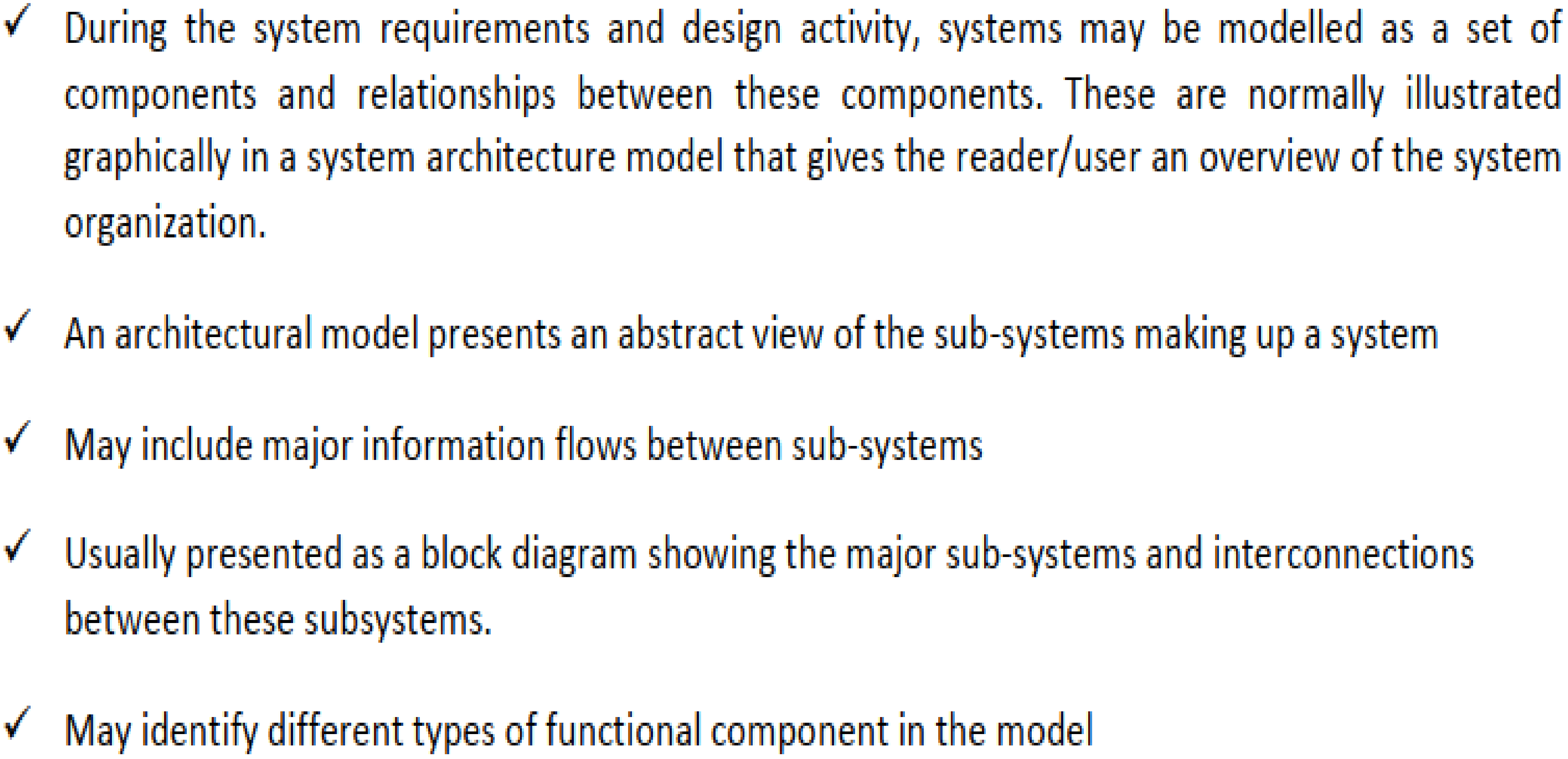
7. System decommissioning

✓ It means taking the system out of service after its useful lifetime

✓ May require data to be restructured and converted to be used in some other system

✓ If the data in the system that is being decommissioned is still valuable to your organization or you may have to convert it for use by other systems.

System modelling



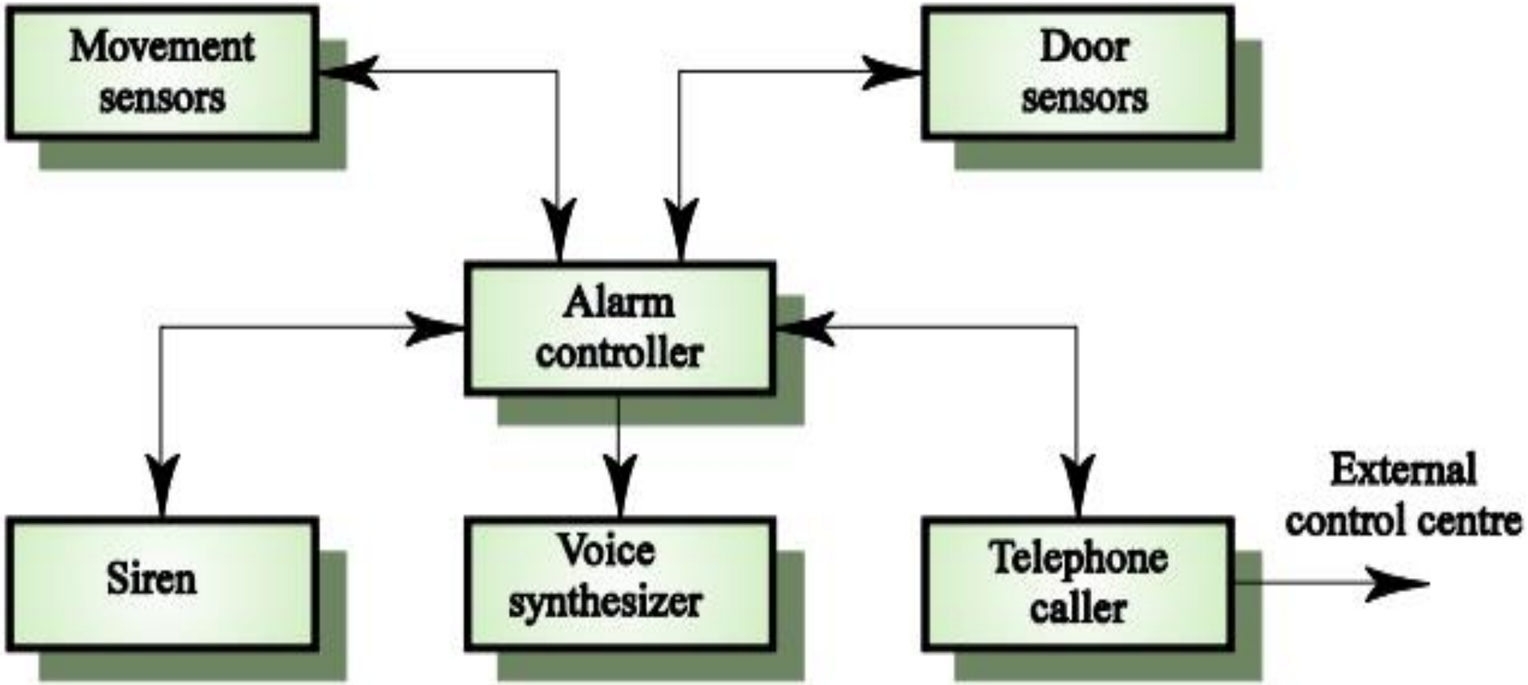
✓ During the system requirements and design activity, systems may be modelled as a set of components and relationships between these components. These are normally illustrated graphically in a system architecture model that gives the reader/user an overview of the system organization.

✓ An architectural model presents an abstract view of the sub-systems making up a system

✓ May include major information flows between sub-systems

✓ Usually presented as a block diagram showing the major sub-systems and interconnections between these subsystems.

✓ May identify different types of functional component in the model e.g. Intruder alarm system



Component types in alarm system

- **Movement Sensor:** Detects movement in the rooms monitored by the system
- **Door Sensors:** detects door opening in the external doors of the building
- **Alarm Controller:** Controls the operation of the system
- **Siren:** Emits an audible warning when an intruder is suspected
- **Voice synthesizer:** Synthesises a voice message giving the location of the suspected intruder
- **Telephone caller:** makes external calls to notify security, the police, etc.

Classes of Information Systems:

- **Transaction processing systems**
  Transaction processing systems are information system applications that capture and process data about business transactions.

- **Management information systems**
  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

- **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).

- **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.

- **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.

System Analyst:

A systems analyst studies the problems and needs of an organization to determine how people, data, processes, communications, and information technology can best accomplish improvements for the business. When information technology is used, the analyst is responsible for:

- ✓ The efficient capture of data from its business source,
- ✓ The flow of that data to the computer,
- ✓ The processing and storage of that data by the computer, and
- ✓ The flow of useful and timely information back to the business and its people.

Task of System Analyst:

1. Identify the problem.

2. Analyze and understand the problem.

3. Identify solution requirements or expectations.

4. Identify alternative solutions and decide a course of action.

Attributes of System analyst:

1. Working Knowledge of Information Technology

- ✓ The systems analyst is an *agent of change*.
- ✓ The systems analyst is responsible for showing end-users and management how new technologies can benefit their business and its operations.
- ✓ The systems analyst must be aware of both existing and emerging information technologies and techniques.

2. Programming Experience and Expertise

- ✓ A systems analyst must know how to program because they are the principle link between business users and computer programmers.
- ✓ It is wrong to assume that a good programmer will become a good analyst or that a bad programmer could not become a good analyst.
- ✓ Most systems analysts need to be proficient in one or more high-level programming languages.

3. Computer Programming Experience and Expertise

- ✓ Historically, the language of choice has been COBOL for business applications, but many organizations are shifting to **visual programming languages** or to **object-oriented programming languages.**
- ✓ The reasons for the shift are as follows:
    - The transition to graphical user interfaces.
    - The desire to downsize applications from the mainframe to networks of PCs.
    - The pressures to improve productivity in applications development through rapid, iterative prototyping and the reuse of programming modules called *objects* and *components*.

Visual and object-oriented programming requires a completely different style of program design, construction, and testing.

4. General Business Knowledge

- The systems analysts are expected to immerse themselves in the business and be able to specify and defend technical solutions that address the bottom-line value returned to the business.
- Systems analysts should be able to communicate with business experts to gain knowledge of problems and needs.
- It is not uncommon for systems analysts to develop so much expertise over time they move out of information systems and into the user community.

5. Problem-Solving Skills

- The systems analyst must have the ability to take a large business problem, break that problem down into its component parts, analyze the various aspects of the problem, and then assemble an improved system to solve the problem.
- The systems analyst must learn to analyze problems in terms of causes and effects rather than in terms of simple remedies.

- ✓ The systems analyst must be well organized.
- ✓ System analysts must be able to creatively define alternative solutions to problems and needs.

6. Communications Skills

- ✓ The systems analyst must be able to communicate effectively, both orally and in writing.
- ✓ The systems analyst should have a good command of the English language.
- ✓ Almost without exception, communications skills, not technical skills, prove to be the single biggest factor in career success or failure.

7. Interpersonal Relations Skills

- ✓ Systems work is people-oriented and systems analysts must be extroverted or people-oriented.
- ✓ Interpersonal skills help systems analysts work effectively with people.
- ✓ Interpersonal skills are also important because of the political nature of the systems analyst's job.
- ✓ The systems analyst's first responsibility is to the business, its management, and its workers.
- ✓ The systems analyst must mediate problems between team problems and achieve benefits for the business as a whole.

8. Flexibility and Adaptability

- No two systems development projects encountered by a systems analyst are identical.
- There is no single, magical approach or solution applicable to systems development.
- Successful systems analysts learn to be flexible and adapt to special challenges or situations presented by specific systems development projects.
- The systems analyst must be able to recognize when variations upon (or single-instance exceptions to) development standards are necessary and beneficial to a particular project.
- The systems analyst must be aware of the implications of not following the standards.

9. Character and Ethics

- The nature of the systems analyst's job requires a strong character and sense of ethics.
- **Ethics** is a personal character trait in which an individual(s) understands the difference between 'right' and 'wrong' and acts accordingly.