SE Unit 2

Socio-technical System -

- Include one or more technical systems, but crucially also include knowledge of how the system should be used to achieve some broader objective.
- These systems have defined operational processes, include people as inherent parts of the system, governed by organizational policies and rules and may be affected by external constraints such as national laws and regulatory policies.

Essential Characteristics of socio-technical System -

- They have Emergent properties (Properties of a system as a whole). These properties depend on both system components and the relationships between them and hence they can be evaluated only when the system has been assembled
- They are often Nondeterministic that is when presented with the specific input they may not always produce the same output. Use of the system may create new relationships between the system components and hence change its emergent behavior.
- The extent to which the system supports organizational objectives does not depend on the system but also on the stability of these objectives and the relationships and conflicts between these objectives.

Emergent System Properties -

- Emergent properties are properties of the system as a whole rather than properties that can be derived from the properties of components of a system.
- They are the consequence of the relationships between system components , therefore they can only be assessed and measured once the components have been integrated into a System.

Examples Of Emergent Properties -

Property	Description
Volume	Varies depending on how the component assemblies are arranged and connected
Reliability	Depends on component reliability but unexpected interactions can cause new types of failure and therefore affect the reliability of the system
Security	The security of the system is a complex property that cannot be easily measured. Attacks may be devised that were not anticipated by the system designers and so may defeat built in safeguards
Repair ability	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 the components that are faulty and modify or replace these components
Usability	This property reflects how easy it is to use the system. It depends on the technical system components, its operators and its operating environment

Types Of Emergent Properties -

1. **Functional emergent properties -**

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

2. **Non functional emergent properties -**

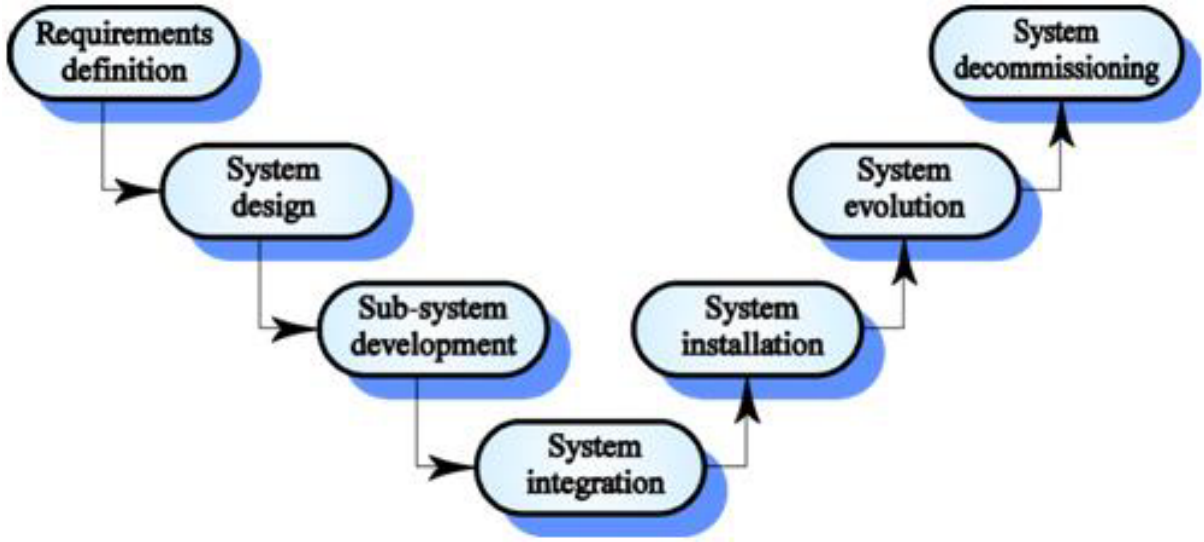
- Relate to behavior of the system in its operational environment These are often critical for computer based systems as failure to achieve some minimal defined level in these properties may make the system unusable.
- Examples are reliability, performance, safety and security.

Systems Engineering -

- Activity of speaking, designing, implementing, validating, deploying and maintaining socio technical systems.
- Systems engineers are not just concerned with software but also with hardware and the systems interactions with users and its environment.
- They think about services that the system provides, the constraints under which the system must be built and operated and the ways in which the system is used to fulfil its purpose

System Engineering Process -

- System Engineering is a discipline which ensures that the customer needs are satisfied throughout the system's life cycle.



- There are important distinctions between the system engineering process and the software development processes.

1. Limited scope for rework during system development: It states that once the system engineering i.e. gathering of ideas and designing has been done in content with the objective and starts with the development, it is difficult to change any requirements.
2. Interdisciplinary involvement: Many different disciplinary approaches has been introduced and integrated as each has different approach and terminology.

Components of system such as organization -

- Socio Technical system are organizational systems intended to help deliver some organizational or business goal.
- If it is difficult to understand the organizational environment where a system is used , the system is less likely to meet the real need of the business and its users.
- Human and organizational factors from the system's environment that affect the system design include -

1. **Process Changes -**

- Does the system require changes to work processes in the environment ?
- If changes happen in the process than training is mandatory.

2. **Job Changes -**

- Does the system de skill the users in an environment or cause them to change the way they work?
- If job changes are resented in the organization ,than the employee has to be well trained to be accepted by the introduction of the system in the organization.

3. **Organizational changes -**

- Does the system change the political power structure in an organization ?
- Depending upon the complexity of the organization ,the changes has to be accepted in the organization.

Legacy Systems -

- Socio-technical computer-based systems that have been developed in the past using an obsolete technology.
- Include not only software and hardware but also legacy processes and procedures-old ways of doing things that are difficult to change because they rely on legacy software.
- Often business critical systems and are maintained because too risky to replace them

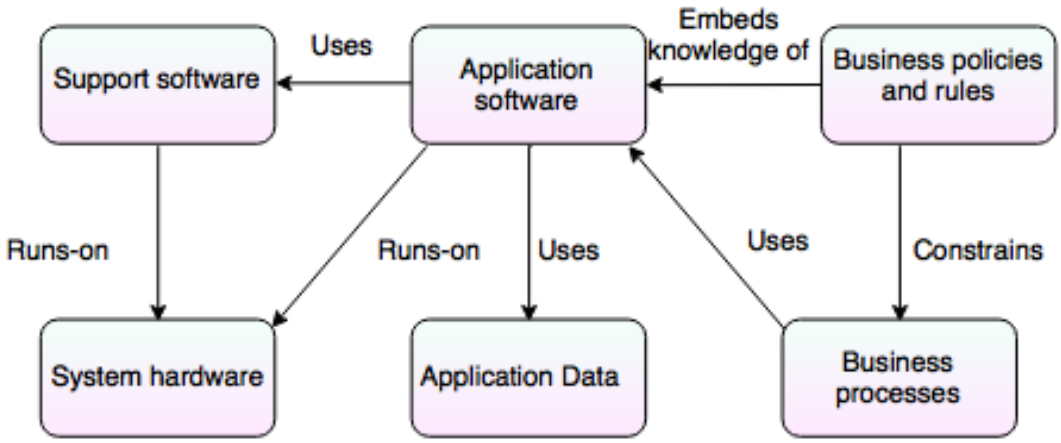
- Legacy systems constrain new business processes and consume a high proportion of company budgets.

Various Issues are -

- Do we throw away and restart or continue to maintain?
- What are the economics (costs and risk) of each approach
- If system depends on other COTS, will upgrades to those be available?

Legacy System Components -

- **Hardware** - may be obsolete mainframe hardware.
- **Support software** - may rely on support software from suppliers who are no longer in business.
- **Application software** - may be written in obsolete programming languages.
- **Application data** - often incomplete and inconsistent
- **Business processes** - may be constrained by software structure and functionality.
- **Business policies and rules** - may be implicit and embedded in the system software.

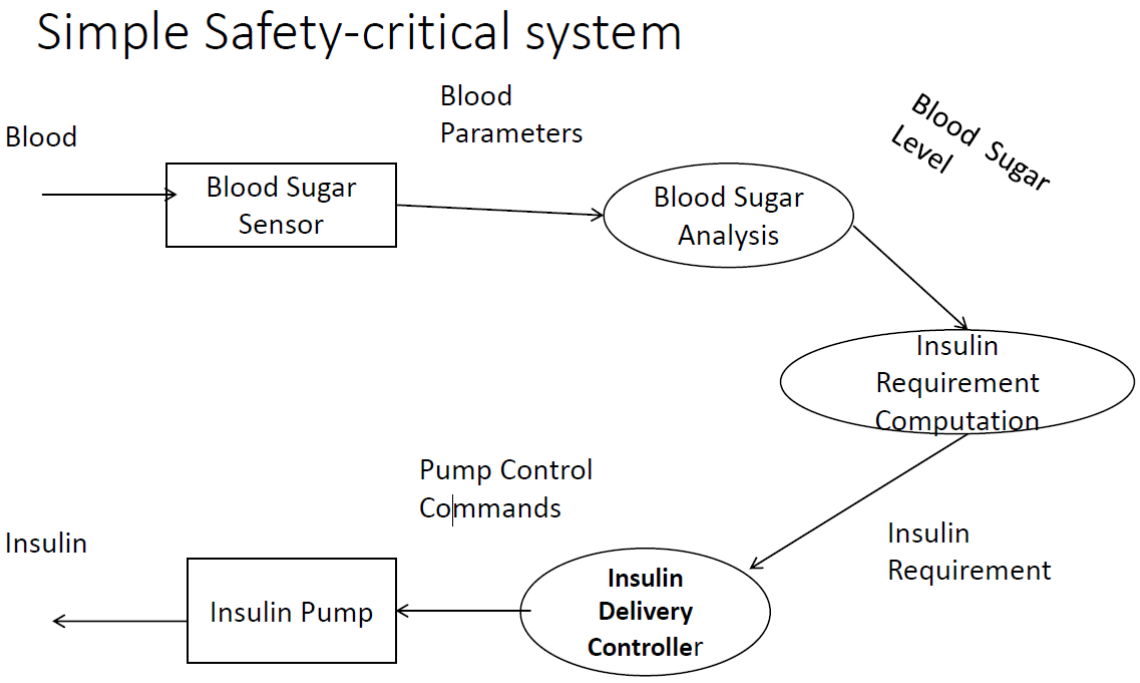


Critical Systems -

- Systems failure that can result insignificant economic losses, physical damage or threats to human life.
- They are technical or socio technical systems that people depend on

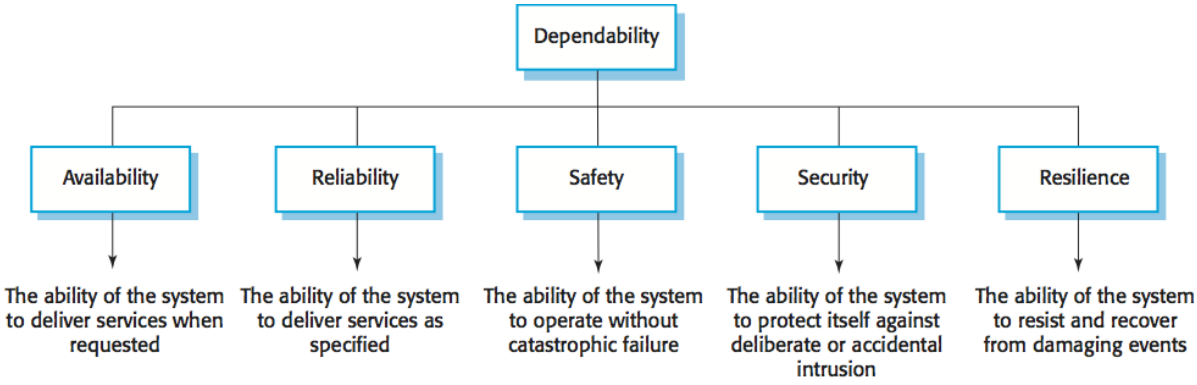
Types Of Critical Systems -

- **Safety-critical system System** - whose failure may result in injury, loss of life or serious environmental damage. Example Control system for a chemical manufacturing plant
- **Mission-critical system System** - whose failure may result in the failure of some goal directed activity. Example Navigational system for a spacecraft
- **Business-critical system System** - whose failure may result in very high costs for the business using that system. Example Customer accounting system in a bank



System Dependability -

The dependability of a computer system is a property equating to its trustworthiness.



- It is the property of the system that equals to its trustworthiness. Four principal dimensions to dependability are -

1. **Availability -** To deliver services when required
2. **Reliability -** To deliver services as specified
3. **Safety -** To operate without terrible failure
4. **Security -** To protect itself against accidental or deliberate interruption

Dependability Specification -

Dependability requirements include -

1. **Functional requirements -** to define error checking and recovery facilities and protection against system failures.
2. **Non functional requirements** - defining the required reliability and availability of the system.
3. **Excluding requirements -** that define states and conditions that must not arise.

A trade off between system performance and system dependability. High dependability can only be achieved at the expense of system performance.

Availability and Reliability -

- System availability and reliability are closely related properties that can both be expressed as numerical probabilities.

- The reliability of a system is the probability that the systems services will be correctly delivered as specified.
- The availability of the system is the probability that the system will be up and running to deliver these services to users when they request them.
- Availability is usually expressed as a percentage of the time that the system is available to deliver services e.g., 99.95%. However, this does not take into account two factors -
- The number of users affected by the service outage. Loss of service in the middle of the night is less important for many systems than loss of service during peak usage periods.
- The length of the outage . The longer the outage, the more the disruption. Several short outages are less likely to be disruptive Than 1 long outage. Long repair times are a particular problem.

Safety -

- These systems that they never damage the people or the systems environment even if the Systems fail like monitoring systems in aircraft etc.
- Safety critical software falls in two classes -
    1. Primary safety critical software Embedded as a controller in a system. Malfunctioning of such software can cause hardware malfunction which results in human injury or environmental damage.
    2. Secondary safety critical software Indirectly results in injury. Example is medical database holding details of drugs administered to patients' error in this could result in wrong dosage of drugs.

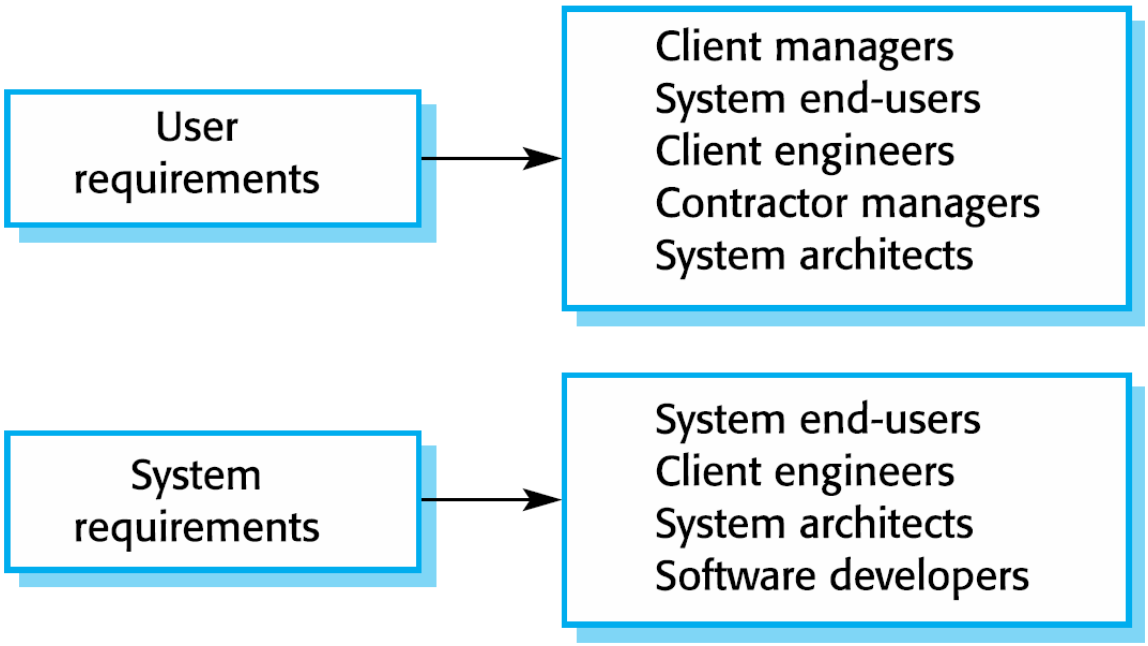
Security -

- Ability of a system to protect itself from external accidental or deliberate attacks. As more systems get connected to Internet it can be attacked by people with unfriendly intentions.
- Without a reasonable level of security, the availability, reliability and safety of the system may be compromised if external attacks can cause some damage to the system.

Requirement Engineering Processes -

Requirement engineering is the process of establishing the services that the customer requires from a system and the constrains under which it operates and is developed.

Type of Requirements -



Feasibility Studies -

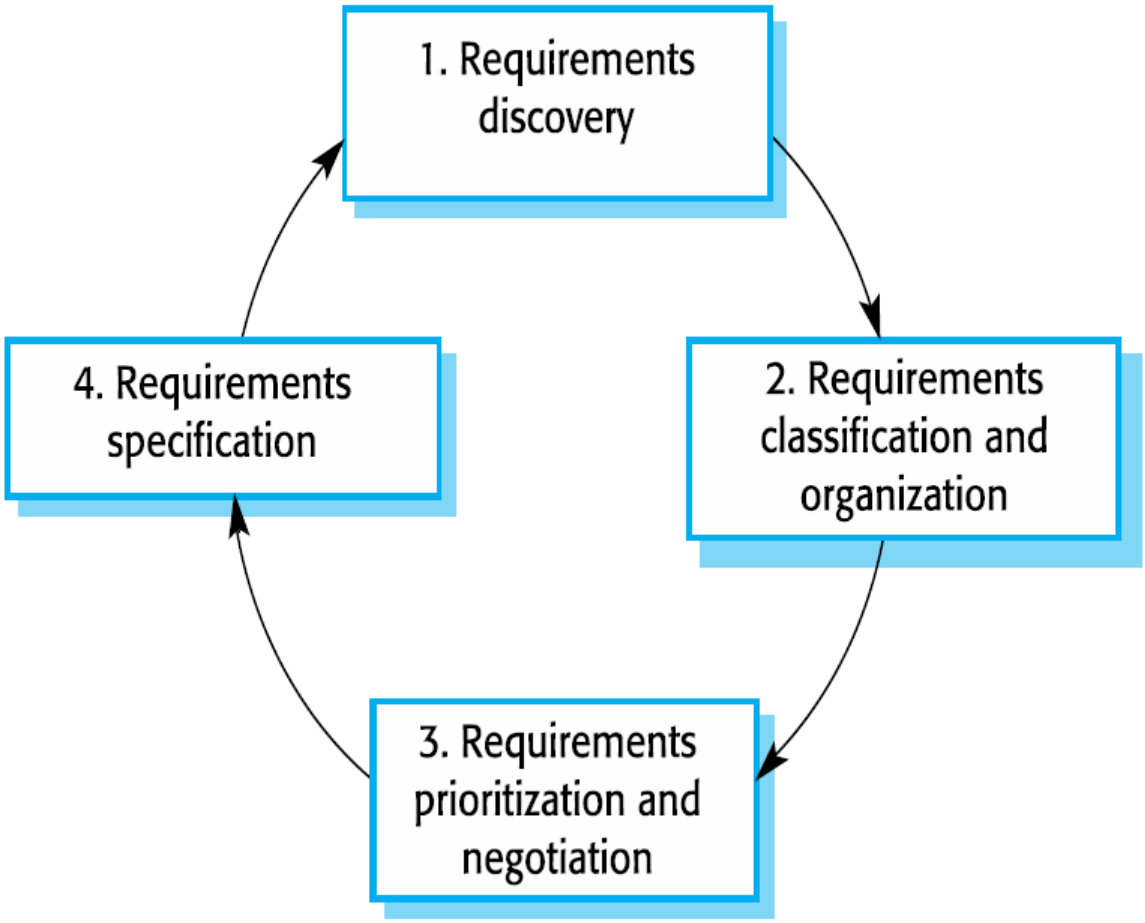
- A feasibility study decides whether or not the proposed system is worthwhile.
- A short focused study that checks -
  - If the system contributes to organizational objectives;
  - If the system can be engineered using current technology and within budget;
  - If the system can be integrated with other systems that are already used.
- Based on information assessment (what is required), information collection and report writing.

- Questions for people in the organization -
  - What if the system wasn’t implemented?
  - What are current process problems?
  - How will the proposed system help?
  - What will be the integration problems?
  - Is new technology needed? What skills?
  - What facilities must be supported by the proposed system?

Requirements Elicitation and Analysis -

- Sometimes called requirements elicitation or requirements discovery.
- Involves technical staff working with customers to find out about the application domain, the services that the system should provide and the system’s operational constraints.
- May involve end users, managers, engineers involved in maintenance, domain experts, trade unions, etc. These are called stakeholders.

The Requirements Elicitation and Analysis Process -



Requirement Validation Checks -

- **Validity che**cks - A system is needed to perform certain functions
- **Consistency checks** - Requirements should not be contradictory or of the same system function
- **Completeness checks** - requirements document should include all functions and constraints
- **Realism checks** - Requirements should be checked to ensure that they could actually be implemented
- **Verifiability** - Requirements should always be written so that they are verifiable

Requirements Validation -

- It is concerned with showing that the requirements actually define the system that the customer wants
- It is important because errors in requirements documentation can lead to extensive rework costs when they are discovered during development process
- The cost of fixing a requirements problems is much greater than repairing design or coding errors

Requirements checking -

- Validity. Does the system provide the functions which best support the customer’s needs?
- Consistency. Are there any requirements conflicts?
- Completeness. Are all functions required by the customer included?
- Realism. Can the requirements be implemented given available budget and technology
- Verifiability. Can the requirements be checked?

Requirements Management -

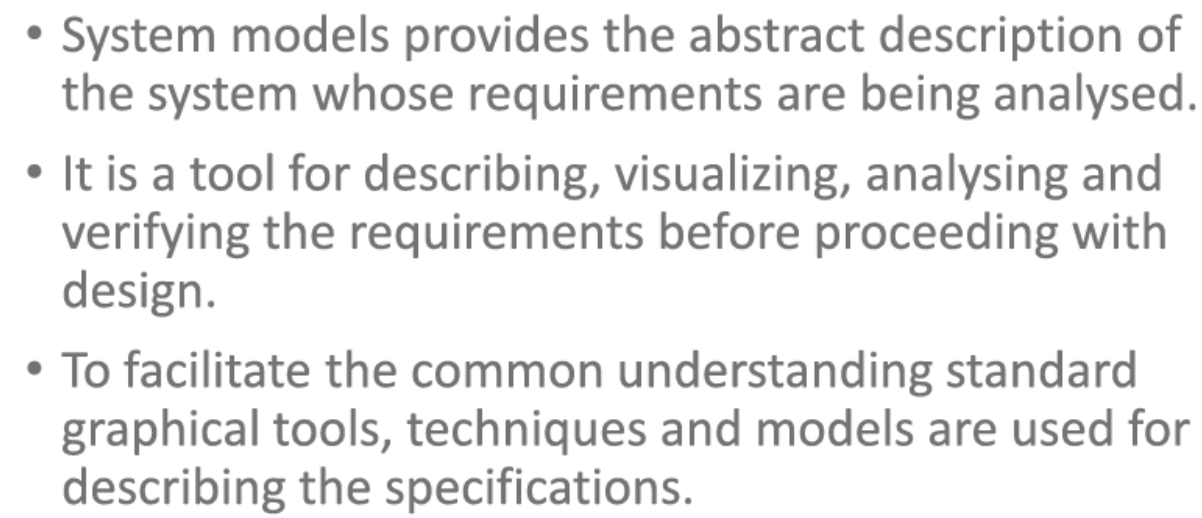
- Requirements management is the process of managing changing requirements during the requirements engineering process and system development.
- New requirements emerge as a system is being developed and after it has gone into use.
- You need to keep track of individual requirements and maintain links between dependent requirements so that you can assess the impact of requirements changes. You need to establish a formal process for making change proposals and linking these to system requirements.

Requirement Management Planning -

- Establishes the level of requirements management detail that is required.
- Requirements management decisions -

- **Requirement's identification** Each requirement must be uniquely identified so that it can be cross referenced with other requirements.
- **A change management process** This is the set of activities that assess the impact and cost of changes.
- **Traceability policies** These policies define the relationships between each requirement and between the requirements and the system design that should be recorded.
- **Tool support** Tools that may be used range from specialist requirements management systems to spreadsheets and simple database systems.

System Model -



Essential elements of system models are -

1. **Environmental Model** - It defines the scope of the proposed system and its boundaries. It consist of statement of purpose, context diagram and events of the system.
2. **Behavioral Model** - It describes the functional requirements, internal behavioral and data entities of the system. It consists of ER diagram, DFD, State Transition diagram.
3. **Implementation Model** - It describes the design specification of the software and consist of software architecture, data design, interface design and

component design.

4. **Structural Model** - It emphasize on modelling the structure of the data that is processed by the system.

Types of System Model -

- **Data processing Model** - Data processing model showing how the data is processed at different stages.
- **Composition Model** - Composition model showing how entities are composed of other entities.
- **Architectural Model** - Architectural model showing principal subsystems
- **Classification Model** - Classification model showing how entities have common characteristics.
- **Stimulus/response model** - Stimulus/response model showing the system's reaction to events.

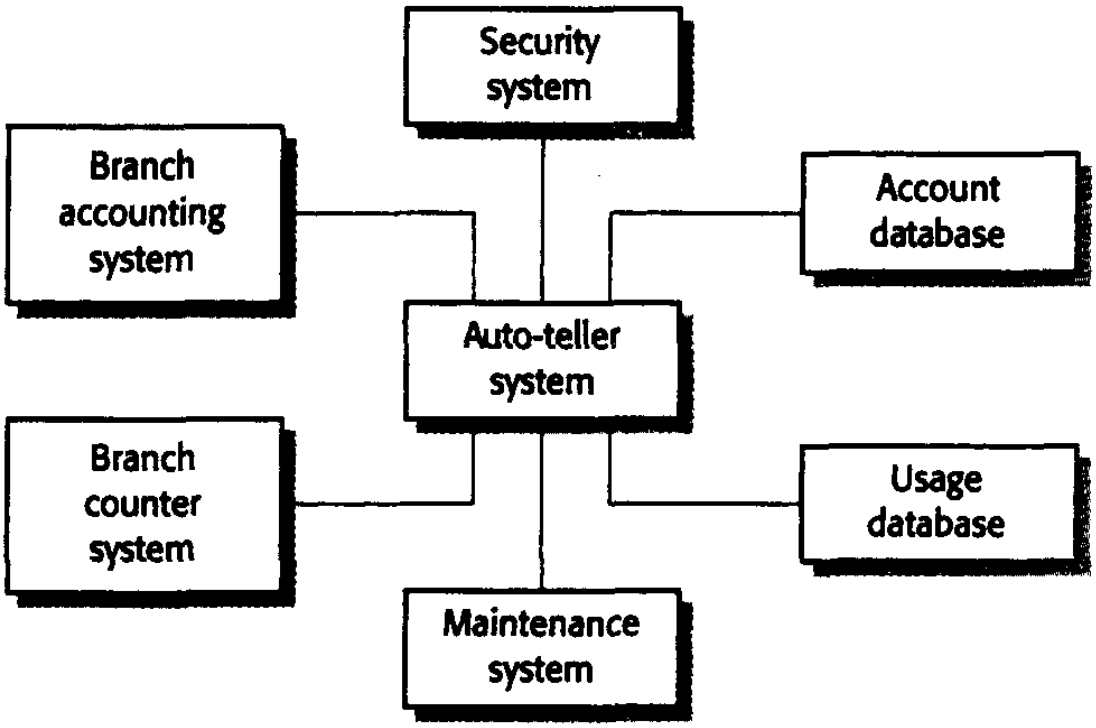
Context Models -

- Context models are used to illustrate the boundaries of a system
- Social and organizational concerns may affect the decision on where to position system boundaries
- Architectural models show the a system and its relationship with other systems

Context Of ATM System -

- Behavioral models are models of the dynamic behavior of a system as it is executing. They show what happens or what is supposed to happen when a system responds to a stimulus from its environment.
  •
  You can think of these stimuli as being of two types:
  •
  Data Some data arrives that must be processed by the system.
  •

Events Some event happens that triggers system processing. Events may have associated data, although this is not always the case.



- In the above figure each ATM is connected to account database, local branch accounting system, a security system and a system to support machine maintenance.
- The system is also connected to usage database that monitors how the networks of ATM is used and to a local branch counter system.
- This counter system provides services such as backup and printing.
- These therefore need not be included in the ATM system itself
- Architectural models describes the environment of the system but do not show the relationships between the other systems in the environment and the system that is being specified
- Simple architecture models are supplemented by other models such as process models that show the process activities by the system

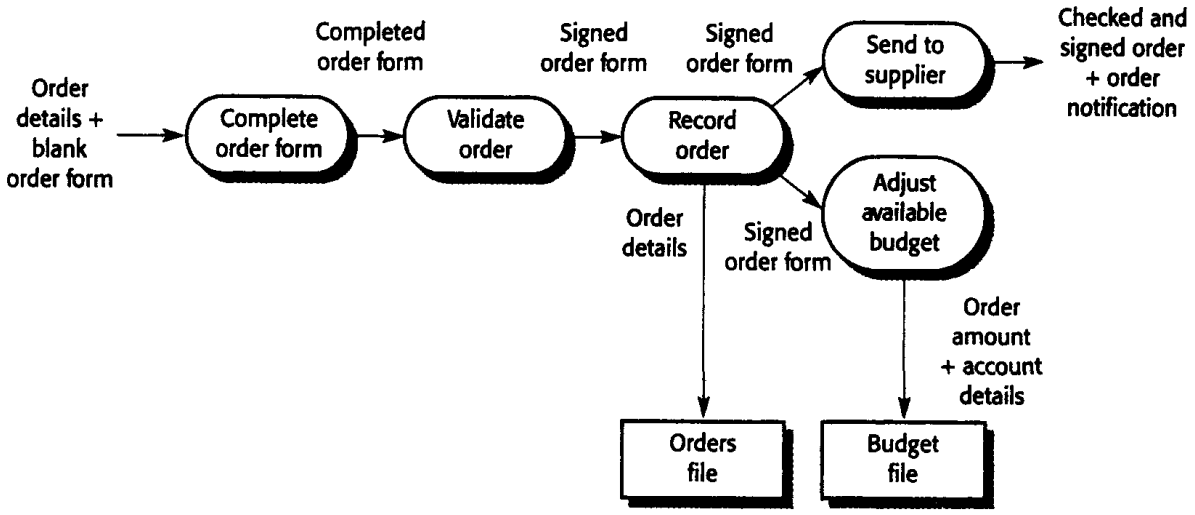
Behavioral Model -

- Behavioral models are models of the dynamic behavior of a system as it is executing. They show what happens or what is supposed to happen when a system responds to a stimulus from its environment.
- You can think of these stimuli as being of two types:
  - Data Some data arrives that must be processed by the system.
  - Events Some event happens that triggers system processing. Events may have associated data, although this is not always the case.

Data Models -

- An important part of system modeling is defining the logical form of data processed by the system. These are called as semantic data models.
- The most widely used data modeling technique is ERA (Entity Relation Attribute) modeling
- The relationship models devised from this system are in 3NF and hence they been widely used
- Data models lack detail and more descriptions of ERA must be maintained.
- Data dictionaries are used to develop system models
- It is simply an alphabetical list of names included in model
- The advantages of data dictionary are it checks for the uniqueness and warns against name duplications and it stores all data in a single place
- The following figure is an example of data model

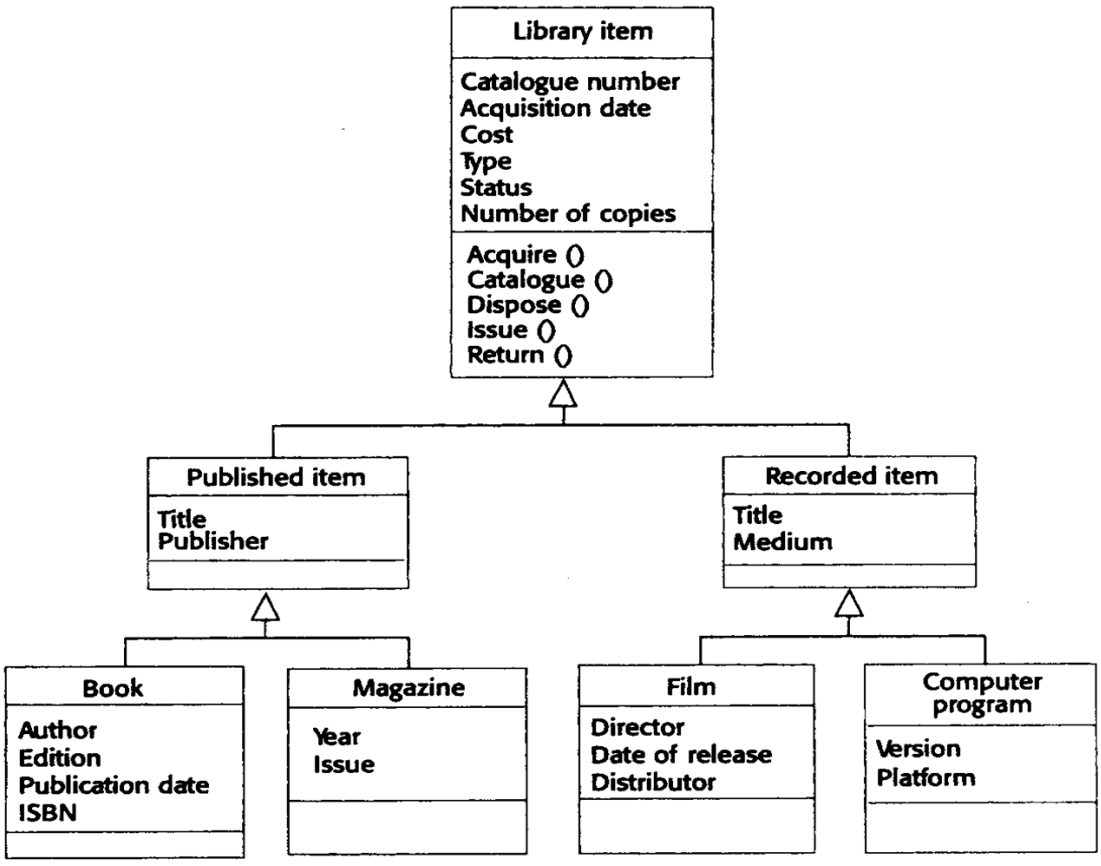
Data Flow Model For Order Processing -



Object Model -

- Expressing the system requirements using object model, designing using objects and developing using languages like C++ and Java
- Object models developed during requirements analysis are used to represent both data and its process. They combine some uses of dataflow and semantic models
- They are also useful for showing how entities in the system may be classified and composed of other entities. Objects are executable entities with attributes and services of the object class and many objects can be created from a class
- The following diagram shows an object class in UML as a vertically oriented rectangle with three sections – name of the object, class attributes, operations associated with the object

---



Structured Methods -

- Systematic way of producing models of an existing system or proposed system

- Provide a framework for detailed system modeling as part of requirements elicitation and analysis
- Have their own preferred set of system models and usually define a process that are used to derive these models and set of rules and guidelines that apply to the models
- CASE tools usually used support model editing, coding, report generation and some model checking capabilities
- Have been applied in many large projects because they use standard notations and ensure standard design documentation
- Suffer from following weakness -
    - Do not provide effective support for understanding non-functional requirements
    - Do not include guidelines whether a method is appropriate nor do they advice on how they can be adapted for a particular project
    - Produce too much documentation
    - Models produced are very detailed hence difficult to understand