

BACKGROUND..



- ✓ What is a Requirement?
 - A condition or capability that must be possessed by a system (IEEE)
- ✓ The requirements task:
 - Input: User needs in minds of people
 - Output: precise statement of what the future system will do
- ✓ What is the output of the Req. phase ?
 - A **software requirements specification (SRS) document**
- ✓ What is an SRS ?
 - A complete specification of what the proposed system should do!

BACKGROUND..



- ✓ Requirements understanding is hard
 - Visualizing a future system is difficult
 - Capability of the future system not clear, hence needs are not clear
 - Requirements change with time
 - ...
- ✓ It is essential to do a proper analysis and specification of requirements

PURPOSE OF SRS DOCUMENT



- ✓ SRS establishes basis of agreement between the **user** and the **supplier**.
 - Users needs have to be satisfied, but user may not understand software
 - Developers will develop the system, but may not know about problem domain (business cases)
- ✓ SRS is
 - the medium to bridge the communications gap, and
 - specifies user needs in a manner both can understand

NEED FOR SRS...



- ✓ Helps user understand his needs.
 - users do not always know their needs
 - must analyze and understand the business problem
 - The requirement process helps clarify needs
- ✓ SRS provides a reference for validation of the final product
 - Clear understanding about what is expected.
 - Validation - “ SW satisfies the SRS “

NEED FOR SRS...

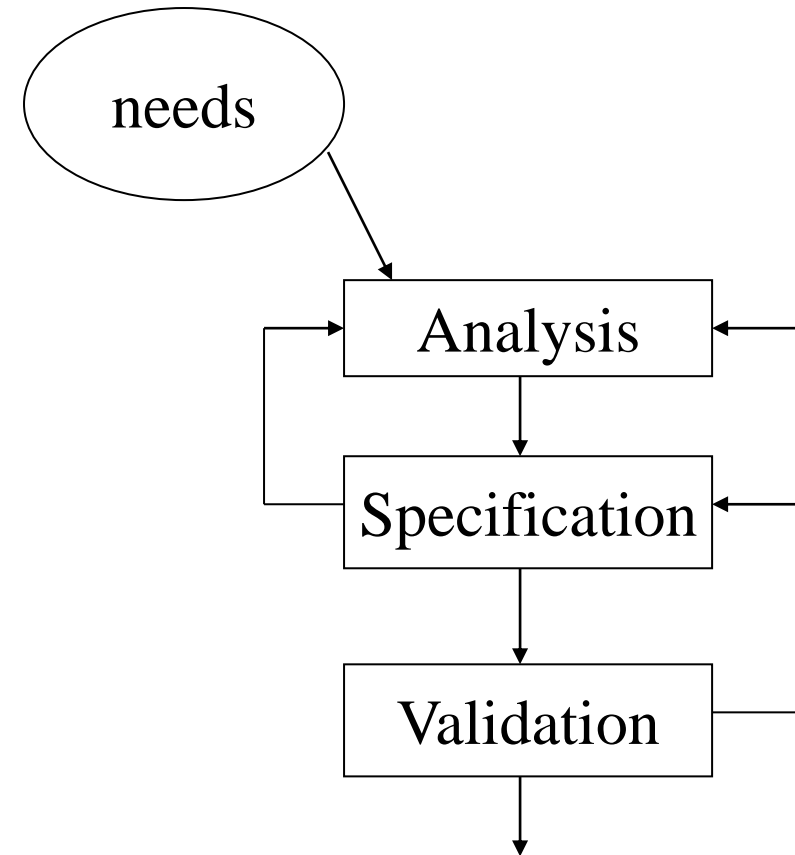


- ✓ High quality SRS is essential for high Quality SW
 - Requirement errors get manifested in final SW
 - To satisfy the quality objective, must have high quality SRS
 - Requirements defects cause later problems
 - 25% of all defects in one study; 54% of all defects found after user testing
 - defects often found in previously approved SRS.
- ✓ Good SRS reduces the development cost
 - SRS errors are expensive to fix later
 - Req. changes can cost a lot (up to 40%)
 - Good SRS can minimize changes and errors

REQUIREMENT PROCESS



- ✓ Process is not linear, it is iterative and sometime parallel
- ✓ Overlap between phases - some parts may be analyzed and specified
- ✓ Specification itself may help analysis
- ✓ Validation can show gaps that can lead to further analysis and spec



REQUIREMENTS PROCESS...



- ✓ Divide and conquer is the basic strategy
 - Decompose into small parts, understand each part and relation between parts

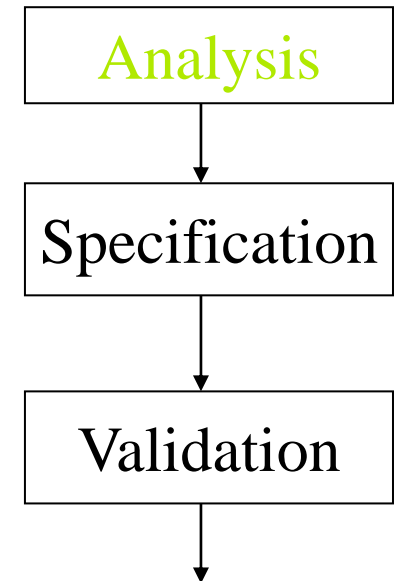
- ✓ Large volumes of information is generated
 - Organizing them is a key

- ✓ Techniques like data flow diagrams, object diagrams etc. used in the analysis

PROBLEM ANALYSIS



- ✓ Aim: to gain an understanding of the needs, requirements, and constraints on the software
- ✓ Analysis involves
 - Interviewing client and users
 - Reading manuals
 - Studying current systems
 - Helping client/users understand new possibilities
 - Like becoming a consultant
- ✓ Must understand the working structure of the organization , client, and users



PROBLEM ANALYSIS..



✓ Some issues

- Obtaining the necessary information
- Brainstorming: interacting with clients to establish desired properties
- Information organization, as large amount of info. gets collected
- Ensuring completeness
- Ensuring consistency
- Basic principle: problem partition
- Partition w.r.t what?
 - Object - OO analysis
 - Function - structural analysis
 - Events in the system – event partitioning

INFORMAL APPROACH TO REQUIREMENT ANALYSIS

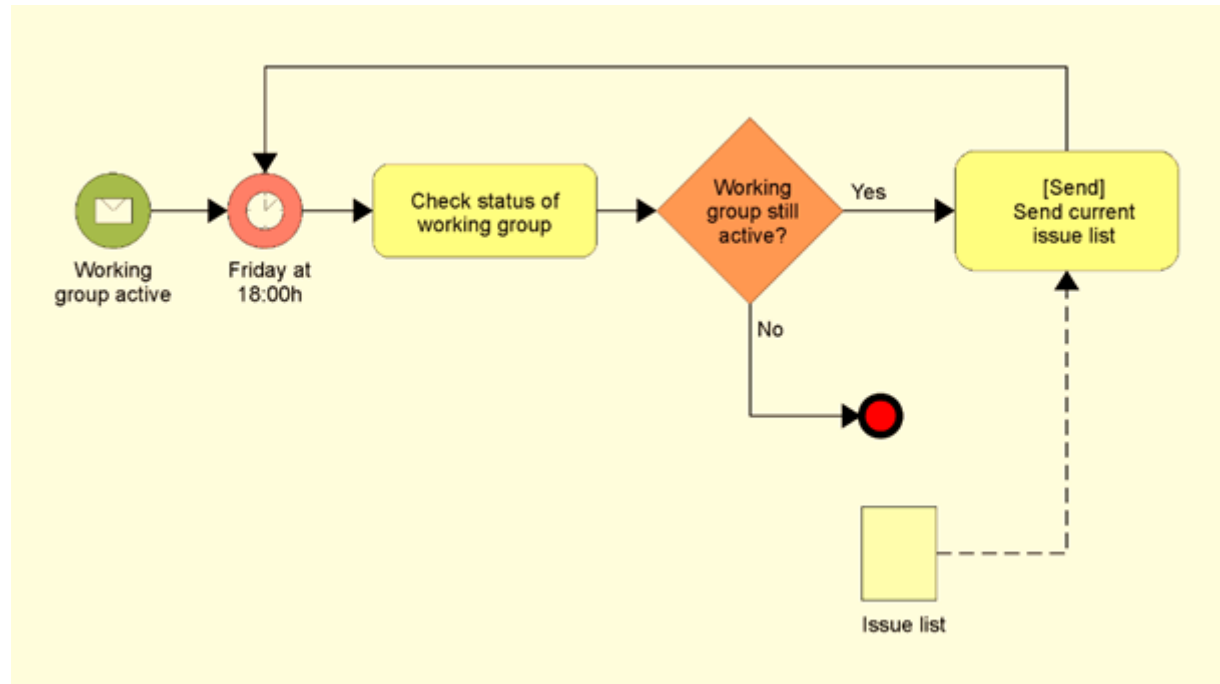


- ✓ No defined methodology; info obtained through analysis, observation, interaction, discussions,...
- ✓ Requirement analysis techniques are mainly used to map the business workflow so that you can analyze, understand and make required changes to that workflow or process.
- ✓ various requirement analyzing techniques that can be used as per the software development process like
 - Business process modeling notation (BPMN)
 - UML (Unified Modeling Language)
 - Flow chart technique
 - Data flow diagram

REQUIREMENT ANALYSIS - BPMN



- ✓ is a graphical representation of business process using simple objects, which helps the organization to communicate in a standard manner. Various objects used in BPMN includes
 - Flow objects
 - Connecting objects
 - Swim lanes
 - Artifacts.
- ✓ should be able to give the detail about the activities carried out during the process like,
 - Who is performing these activities?
 - What data elements are required for these activities?

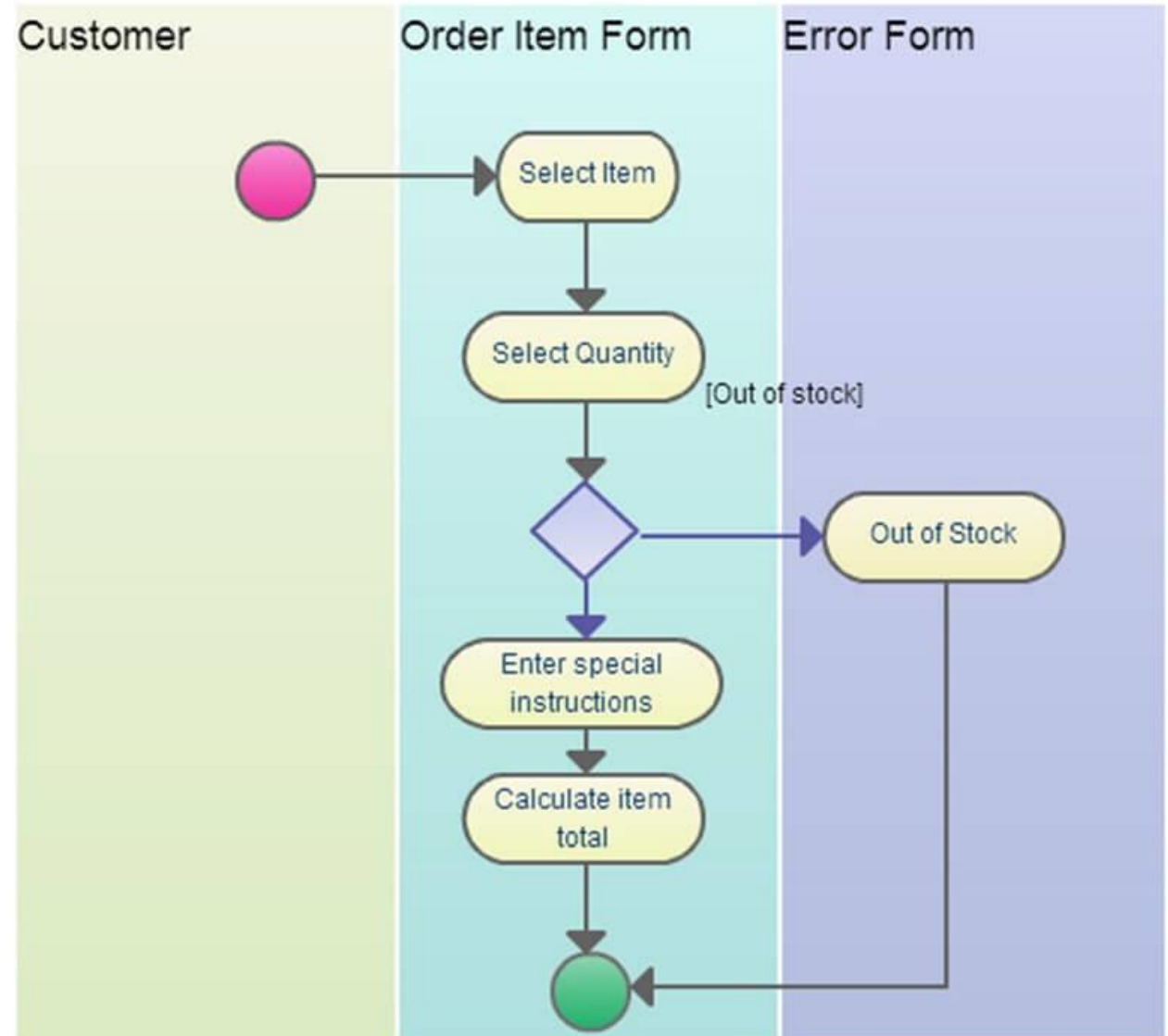


REQUIREMENT ANALYSIS - UML



- ✓ is a modelling standard primarily used for specification, development, visualization and documenting of software system. To capture important business process and artifacts UML provides objects like

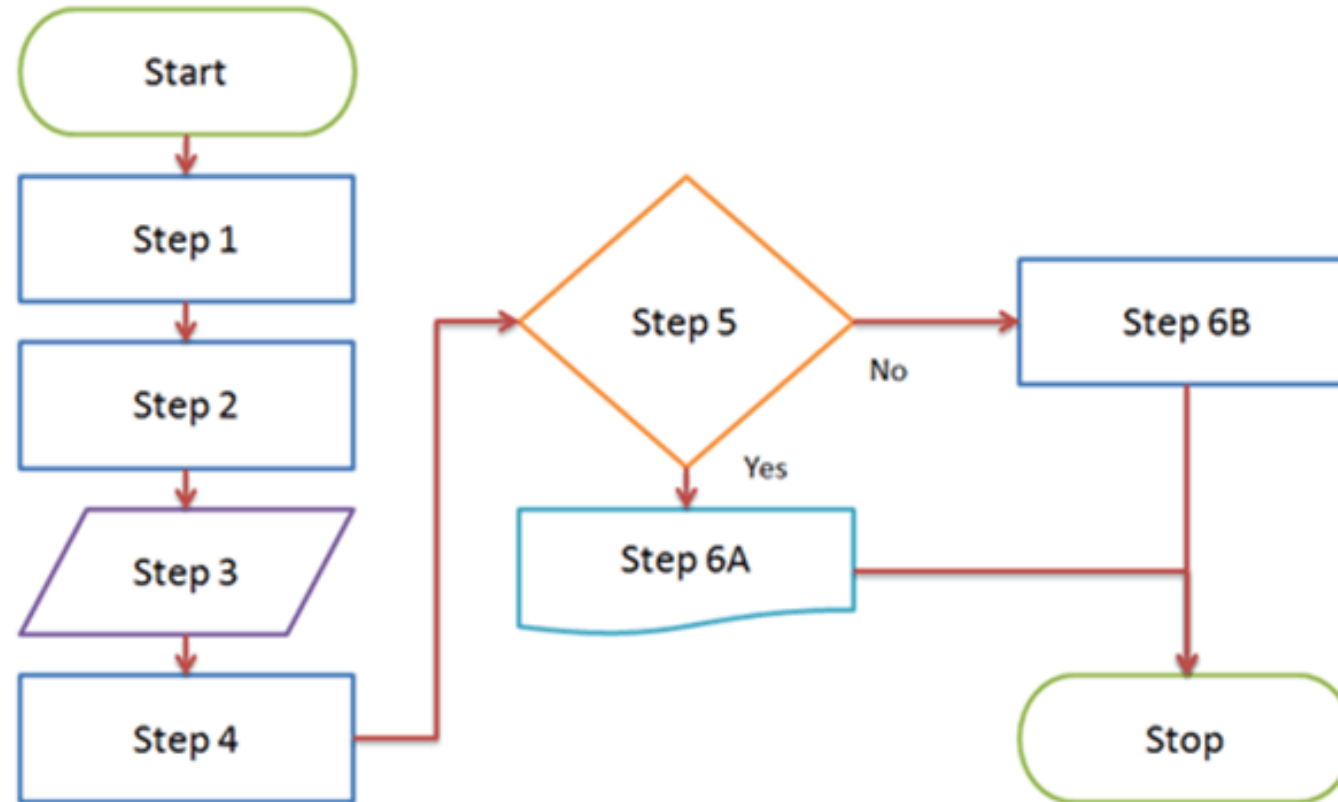
- State
- Object
- Activity
- Class diagram



REQUIREMENT ANALYSIS - FLOWCHART



- ✓ A flowchart is a visual representation of the sequential flow and control logic of a set of related activities or actions.
- ✓ A flow chart can be used for different activities like representing data flows, system interactions, etc.



REQUIREMENT ANALYSIS - DATA FLOW DIAGRAMS

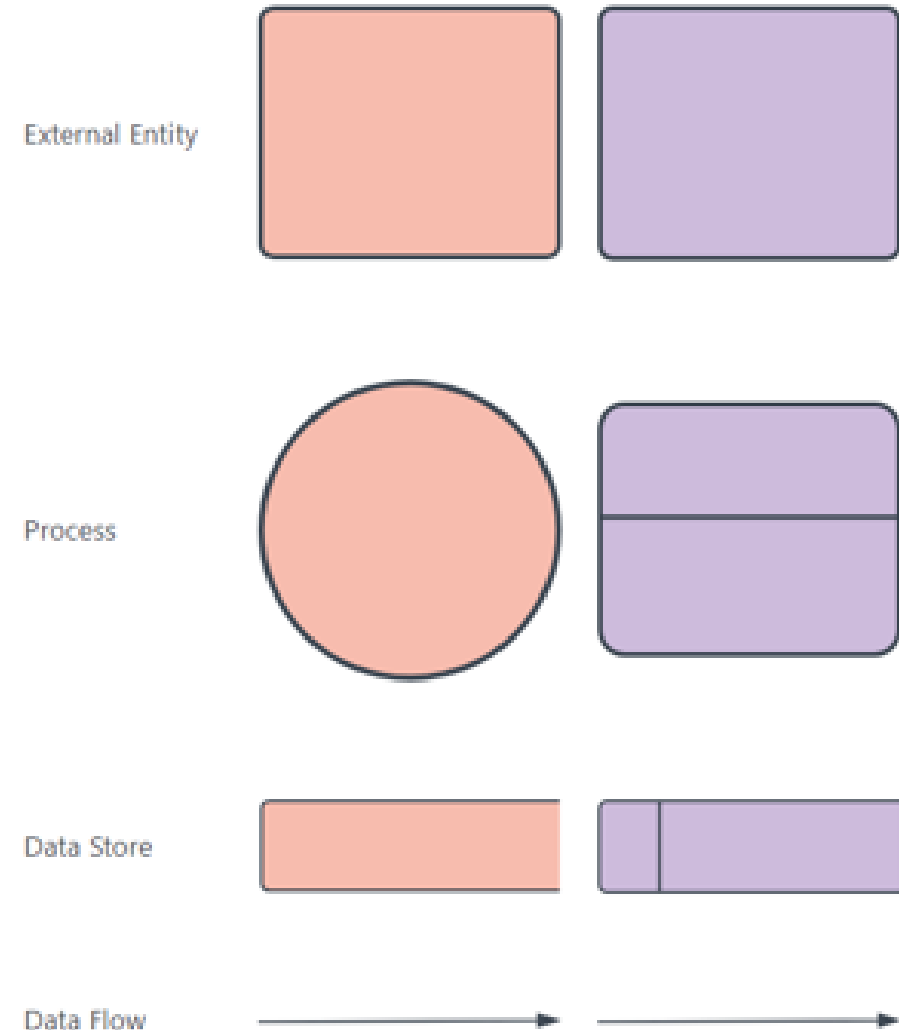


- ✓ Data flow diagrams show how data is processed by a system in terms of inputs and outputs.

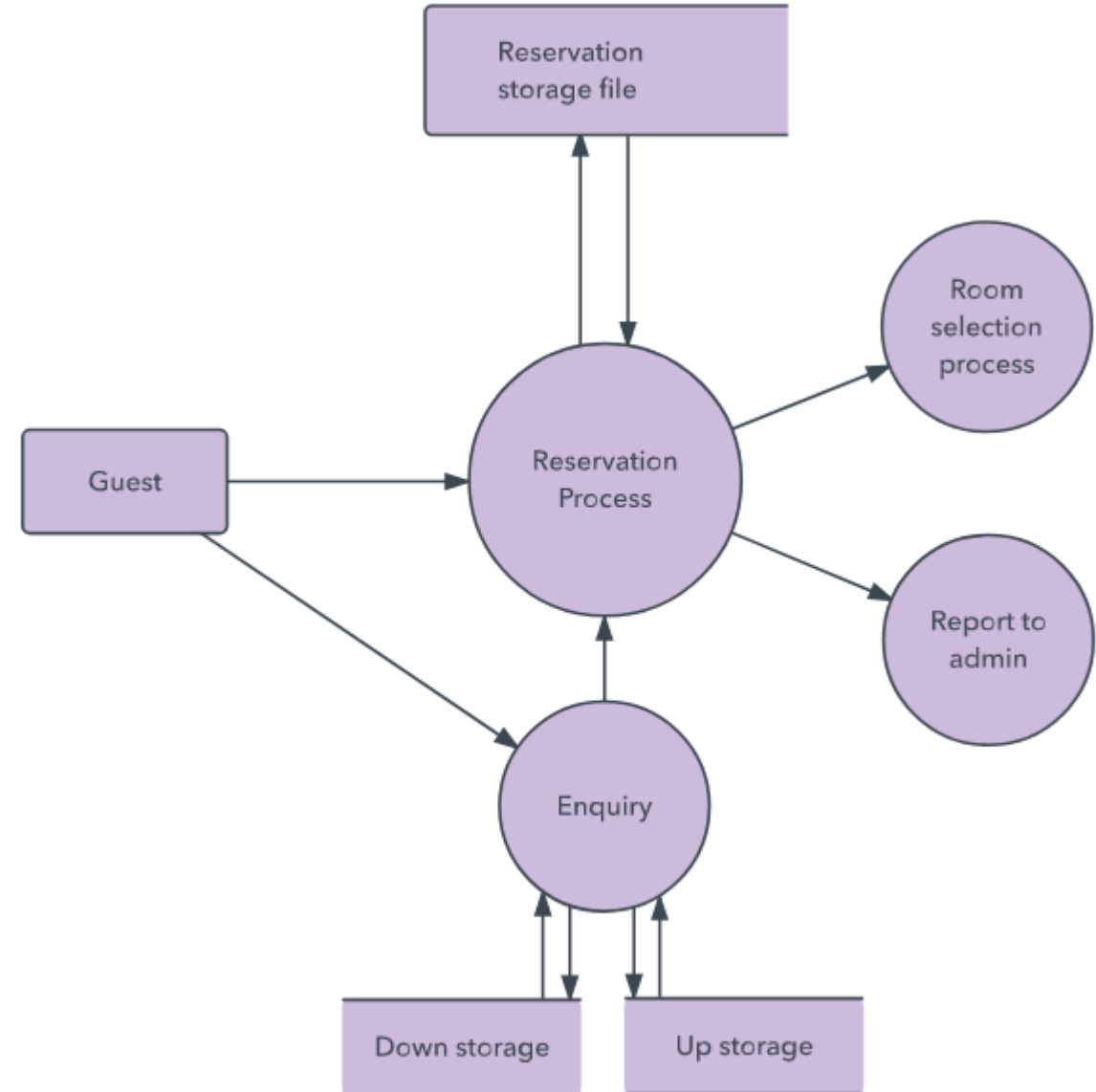
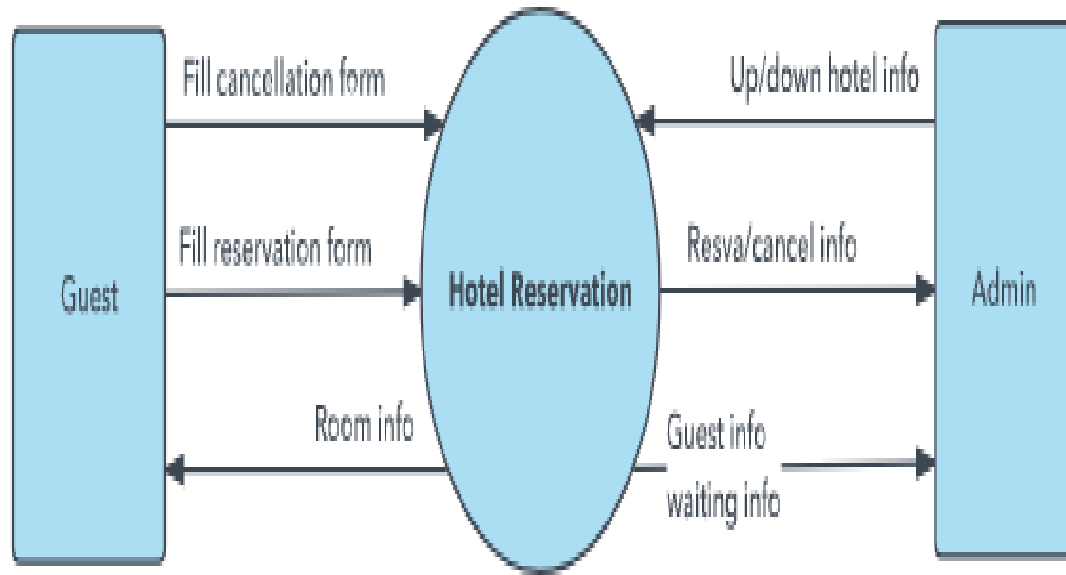
Components of data flow diagram includes

- Process
 - Data Flow
 - Data Store
 - External Entity
- ✓ A DFD shows flow of data through the system
 - Views system as transforming inputs to outputs
 - Transformation done through transforms
 - DFD captures how transformation occurs from input to output as data moves through the transforms

Symbols and Notations Used in DFDs



DFD EXAMPLE



DFD RULES AND TIPS



- ✓ Each process should have at least one input and an output.
- ✓ Each data store should have at least one data flow in and one data flow out.
- ✓ Data stored in a system must go through a process.
- ✓ All processes in a DFD go to another process or a data store.

OTHER APPROACHES TO RA

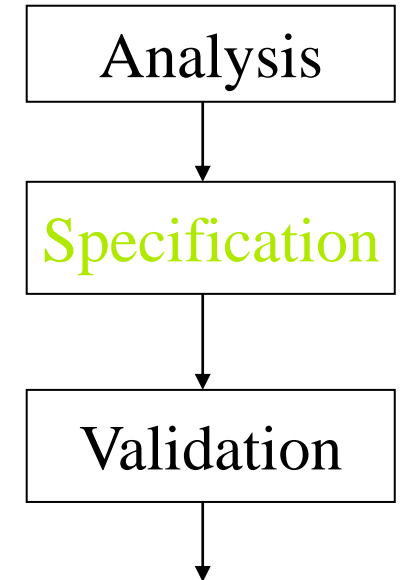


- ✓ Prototyping
- ✓ Object Oriented
 - Classes, attributes, methods
 - Association between classes
 - Class hierarchies

REQUIREMENTS SPECIFICATION



- ✓ Final output of requirements task is the SRS
- ✓ Why are DFDs, OO models, etc not SRS ?
 - SRS focuses on external behavior, while modeling focuses on problem structure
 - UI etc. not modeled, but have to be in SRS
 - Error handling, constraints etc. also needed in SRS
- ✓ Transition from analysis to specification is not straight forward
- ✓ Knowledge about the system acquired in analysis used in specification



"requirement" ≠ "specification"

- ✓ Requirement – understanding between customer and supplier
- ✓ Specification – what the software must do
- ✓ Requirements that are not in the SRS
 - Costs
 - Delivery dates
 - Acceptance procedures
 - etc

COMPONENTS OF AN SRS



Types of Requirements

Functional

- Specifies all the functionality that the system should support
- Outputs for the given inputs and the relationship between them
- All operations the system is to do
- Must specify behavior for invalid inputs too

Design Constraints

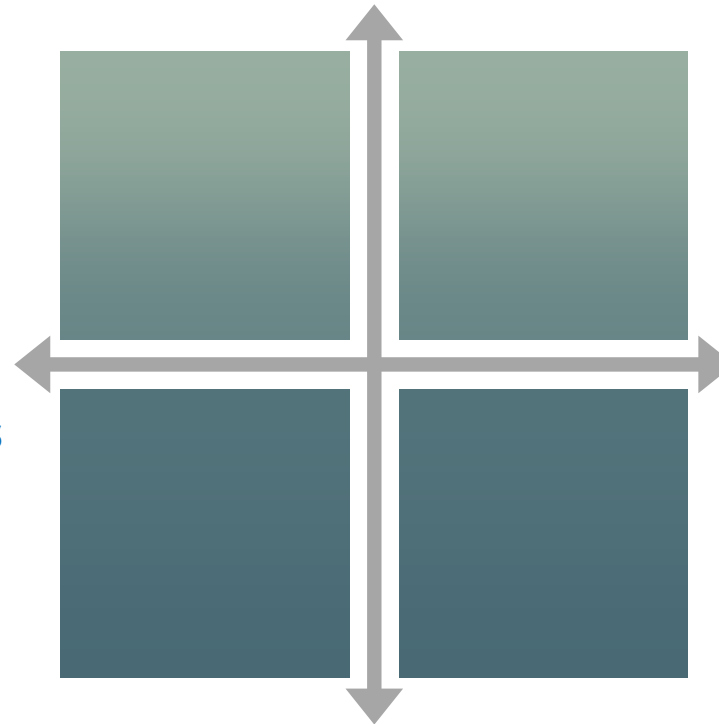
- Factors in the client environment that restrict the choices
- Some such restrictions
 - Standard compliance and compatibility with other systems
 - Hardware Limitations
 - Reliability, fault tolerance, backup req.
 - Security

Non-functional

- All the performance constraints on the software system
- Generally on response time , throughput etc
- Capacity requirements
- Must be in measurable terms

External Interface

- All interactions of the software with people, hardware, and sw
- User interface most important
- General requirements of “friendliness” should be avoided
- These should also be verifiable



CHARACTERISTICS OF SRS



Correctness

- Each requirement accurately represents some desired feature in the final system

Completeness

- All desired features or characteristics are specified
- Completeness and correctness strongly related

Unambiguous

- Each req has exactly one meaning
- Without this errors will creep in
- Important as natural languages often used

Verifiability

- There must exist a cost effective way of checking if sw satisfies requirements

Consistent

two requirements don't contradict each other

Ranked for importance/stability

Needed for prioritizing in construction
To reduce risks due to changing requirements

Traceable

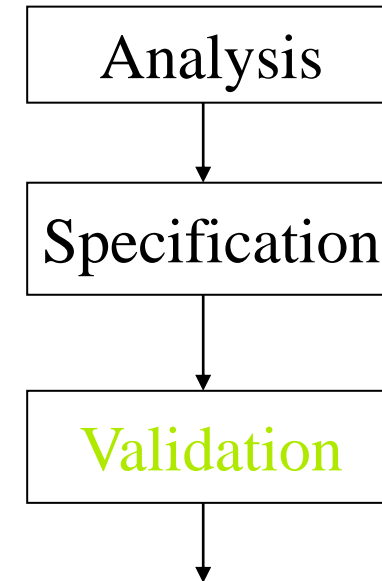
The origin of the req, and how the req relates to software elements can be determined

**SRS
Characteristics**

REQUIREMENTS VALIDATION



- ✓ Lot of room for misunderstanding
- ✓ Errors possible
- ✓ Expensive to fix req defects later
- ✓ Must try to remove most errors in SRS
- ✓ Most common errors
 - Omission - 30%
 - Inconsistency - 10-30%
 - Incorrect fact - 10-30%
 - Ambiguity - 5 -20%



REQUIREMENTS REVIEW



- ✓ SRS reviewed by a group of people
- ✓ Group: author, client, user, dev team rep.
- ✓ Must include client and a user
- ✓ Process – standard inspection process
- ✓ Effectiveness - can catch 40-80% of req. errors