**Lecture: 11**

**Requirement Analysis and Modeling**

**Requirements Modeling:**

Requirements modeling uses a combination of text and diagrammatic forms to depict requirement in a way that is relatively easy to understand and straightforward to review for correctness, completeness and consistency.

A software engineer (or analyst) builds the model using requirements elicited from the customer.

To validate software requirements, one need to examine them from a number of different points of view:

- scenario-based models
  - Scenario-based modeling represents the system from the user's point of view.
- data(information) models
  - Data modeling represents the information space and depicts the data objects that the software will manipulate and the relationships among them.
- class-based models:
  - Class-based modeling defines objects, attributes and relationships.

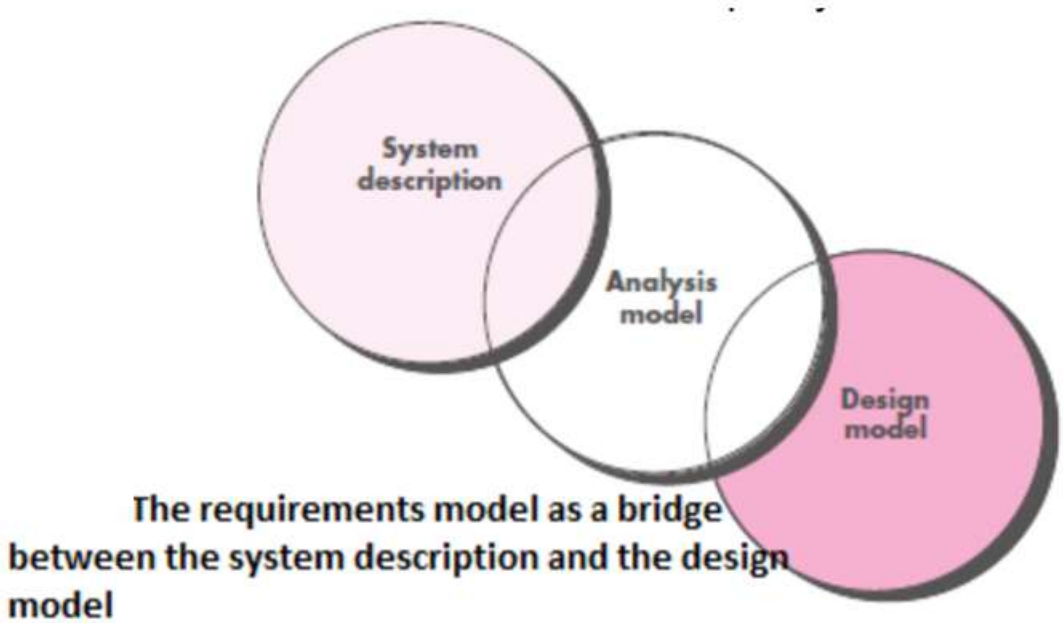
Each represents requirements in a different dimension, thereby increasing the probability that errors will be found, inconsistency will surface and other omissions will be uncovered.

Throughout requirements modeling, the primary focus is **on what and not how**.

The requirements model must achieve three primary objectives:

- to describe what the customer requires.
- to establish a basis for the creation of a software design and
- to define a set of requirements that can be validated once the software is built.

The analysis model bridges the gap between a system-level description that describes overall system or business functionality as it is achieved by applying software, hardware, data, human and other system elements and a software design.




The requirements model as a bridge between the system description and the design model

Arlow and Neustadt suggest a number of worthwhile rules of thumb that should be followed when creating the analysis model:

- The model should focus on requirements that are visible within the problem or business domain.
- Each element of the requirements model should add to an overall understanding of software requirements and provide insight into the information domain, function and behavior of the system.
- Delay consideration of infrastructure and other nonfunctional models until design i.e. a database may be required, but the classes necessary to implement it, the functions required to access it and the behavior that will be exhibited as it is used should be considered only after problem domain analysis has been completed.
- Minimize coupling throughout the system.
- Be certain that the requirements model provides value to all stakeholders.
- Keep the model as simple as it can be. Don't create additional diagrams when they add to new information.