

# **CSC444F: Software Engineering I**

**Mou Hu**  
**[mou.hu@utoronto.ca](mailto:mou.hu@utoronto.ca)**

# Lecture 6: Requirements Engineering (II)

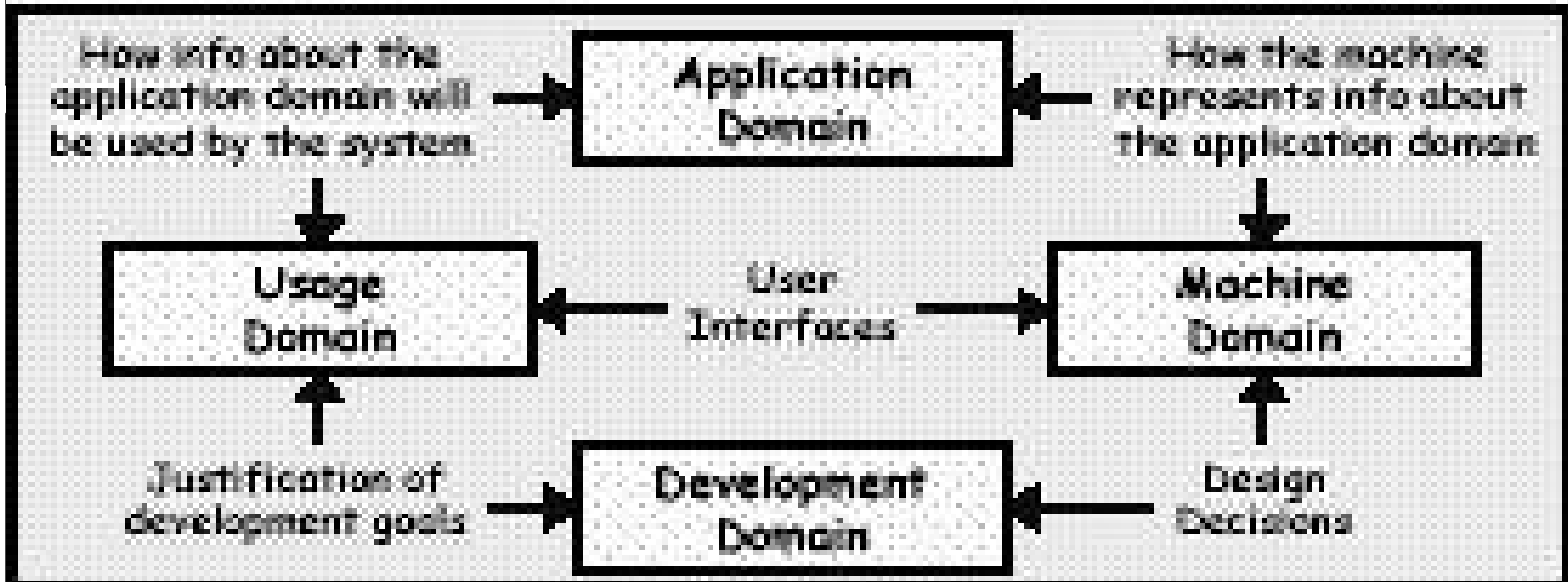
- **Modeling**
- **Requirements Specification Techniques**
  - Entity-Relationship Modeling (ERM)
  - Finite State Machines (FSM)
  - Structured Analysis and Design Technique (SADT)
- **Requirements Specification V & V**
- **Requirements Evolution**
- **Reading: Chapter 9**

# Requirements Specification Techniques

## Modeling: Notations vs. Methods

- **Notation:** a systematic way of presenting something may be linguistic (textual) or graphical (diagrams)
- **A Method provides:** a set of notations (e.g. for different viewpoints); techniques for using those notations (esp. analysis techniques); heuristics to provide guidance
- **Example Methods**
  - **Structured Analysis**  
SADT; SASD; Information Engineering; JSD
  - **Entity-Relationship Approach**
  - **Object Oriented Analysis**  
Coad-Yourdon; OMT; UML
  - **Formal Methods**  
SCR; RSML

# Modeling: Where to Start?



- There are lots of things we could (should) model:
- Structured Analysis starts by modeling the existing system
- Object Oriented Analysis starts by identifying candidate objects

# Structuring the Modeling

## Principle 1: Partitioning

### ■ Partitioning

**captures aggregation/part-of relationship**

### ■ Example:

**goal is to develop a spacecraft**

**partition the problem into parts:**

**guidance and navigation; data handling; command and control; environmental control; instrumentation; etc**

### ■ Note: this is not a design, it is a problem decomposition

**actual design might have any number of components, with no relation to these sub-problems**

**However, the choice of problem decomposition will probably be reflected in the design**

# Structuring the Modeling

## Principle 2: Abstraction

### ■ Abstraction

**A way of finding similarities between concepts by ignoring some details**

**Focuses on the general/specific relationship between phenomena**

- **Classification groups entities with a similar role as members of a single class**
- **Generalization expresses similarities between different classes in an 'is\_a' association**
- **Example: requirement is to handle faults on the spacecraft; might group different faults into fault classes**
- **E.g. based on symptoms of fault or based on location of fault**

# Structuring the Modeling

## Principle 3: Projection

### ■ **Projection:**

separates aspects of the model into multiple viewpoints  
similar to projections used by architects for buildings

### ■ **Example:**

Need to model the communication between spacecraft and ground system

Model separately:

sequencing of messages; format of data packets; error correction behavior; etc.

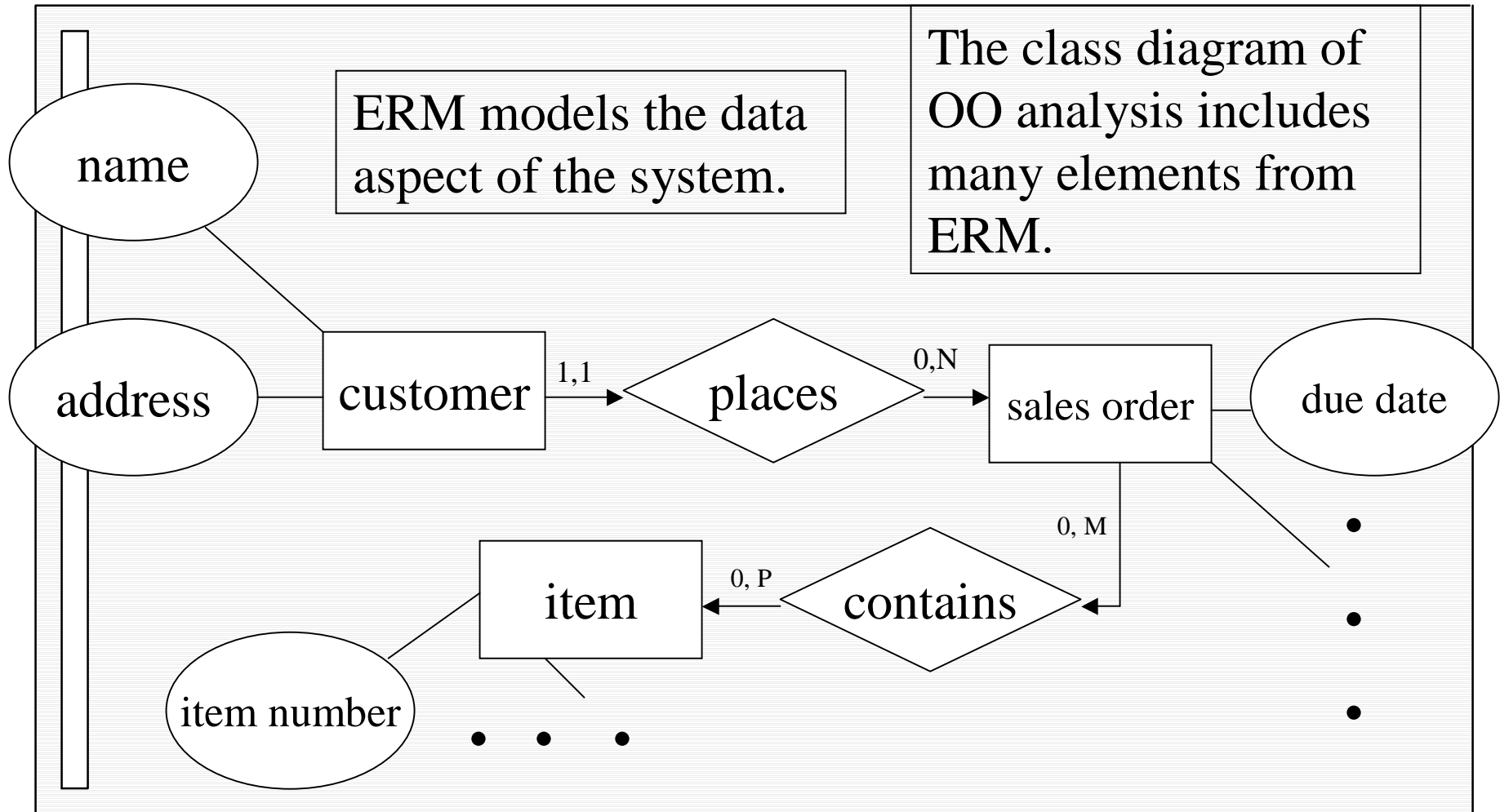
### ■ **Note: Projection and Partitioning are similar.**

Partitioning defines a ‘part of’ relationship

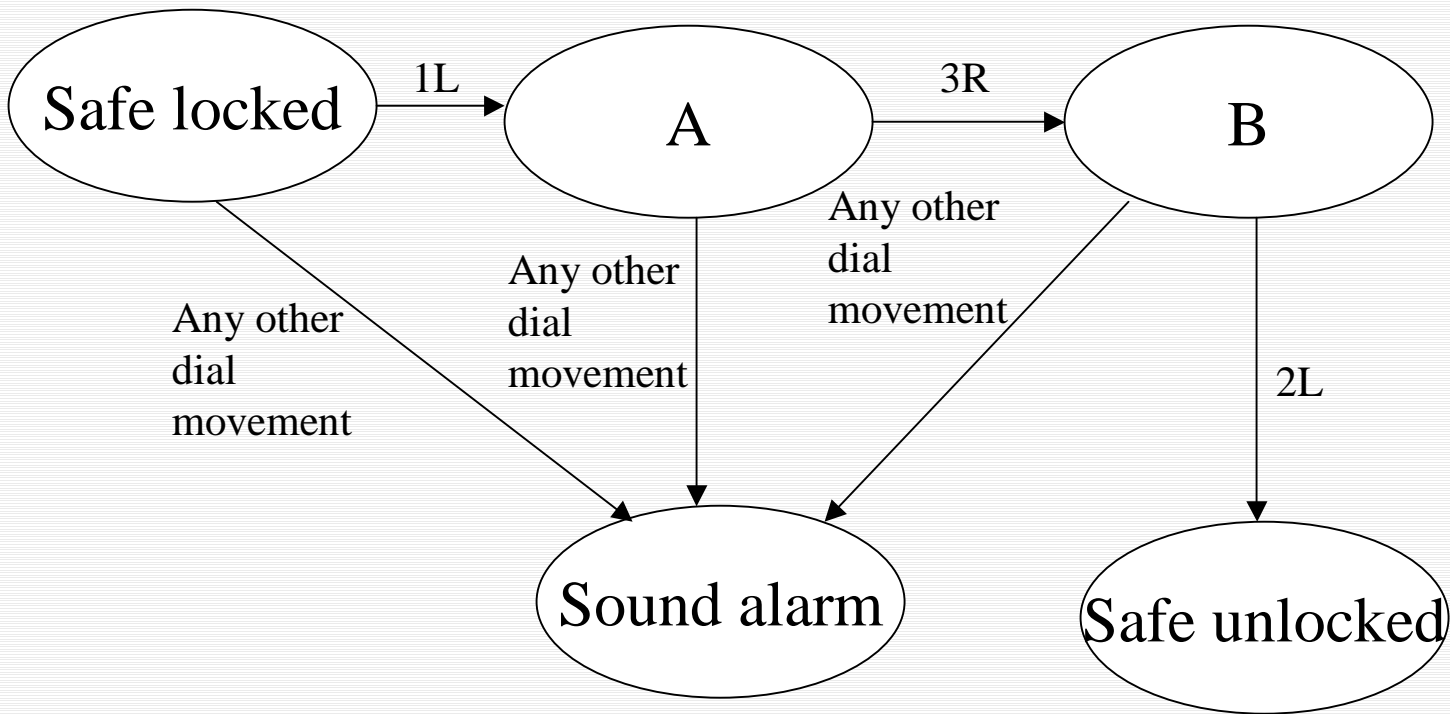
Projection defines a ‘view of’ relationship

Partitioning assumes a the parts are relatively independent

# Entity-Relationship Modeling (ERM)



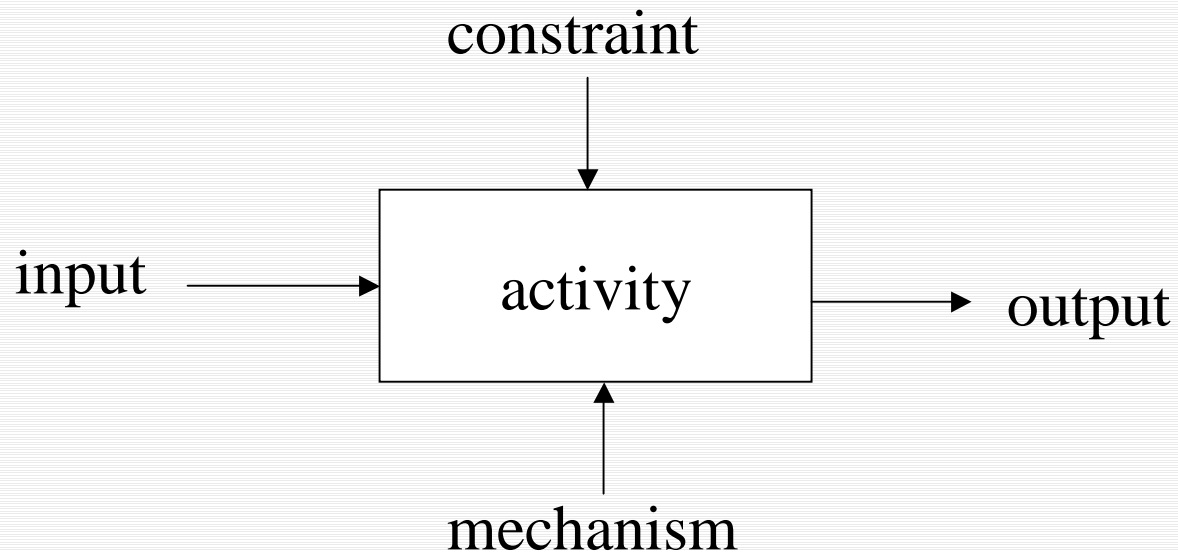
# Finite State Machine (FSM)



FSM models the functional aspect of the system.

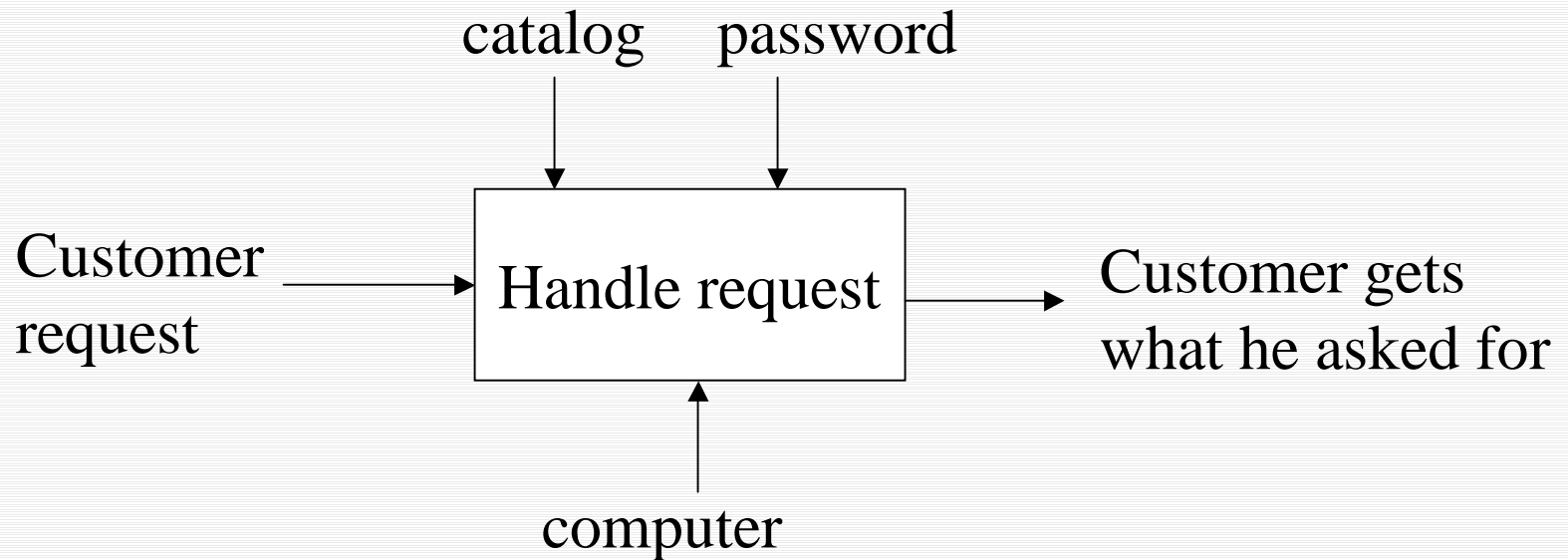
# Structured Analysis and Design Technique (SADT) (1/2)

- SADT describes the functional architecture, based on which a system architecture is built during the design phase.
- The SADT diagrams are constraint diagrams.

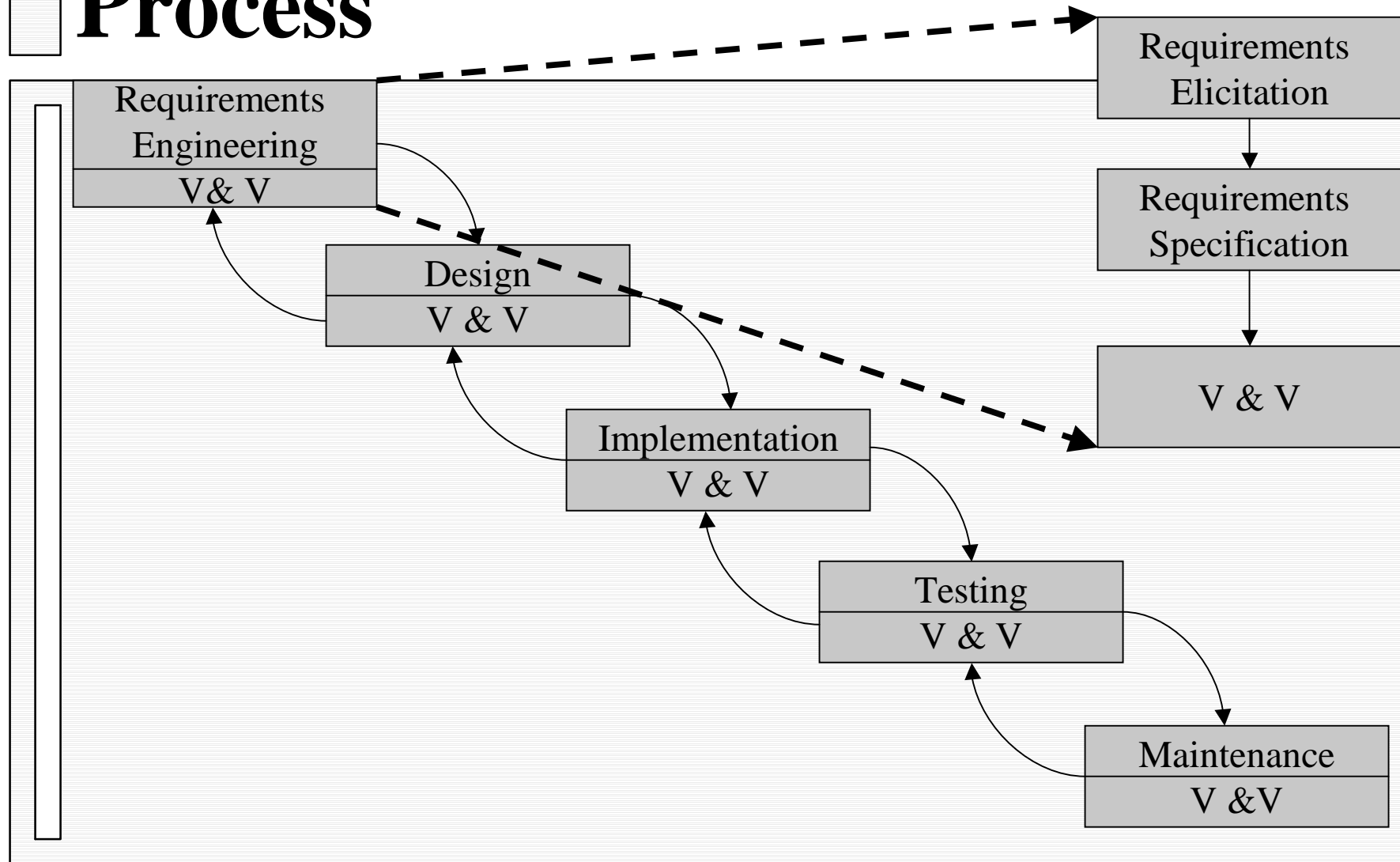


# Structured Analysis and Design Technique (SADT) (2/2)

- A higher level SADT diagram can be decomposed to lower level SADT diagrams. (Textbook Fig. 9.16)



# Basic Requirements Engineering Process



# Requirements Evolution

