

# SOFTWARE DESIGN & ANALYSIS (Week-7)

USAMA MUSHARAF

MS-CS (Software Engineering)

*LECTURER (Department of Computer Science)*

*FAST-NUCES PESHAWAR*

# CONTENTS OF WEEK # 7

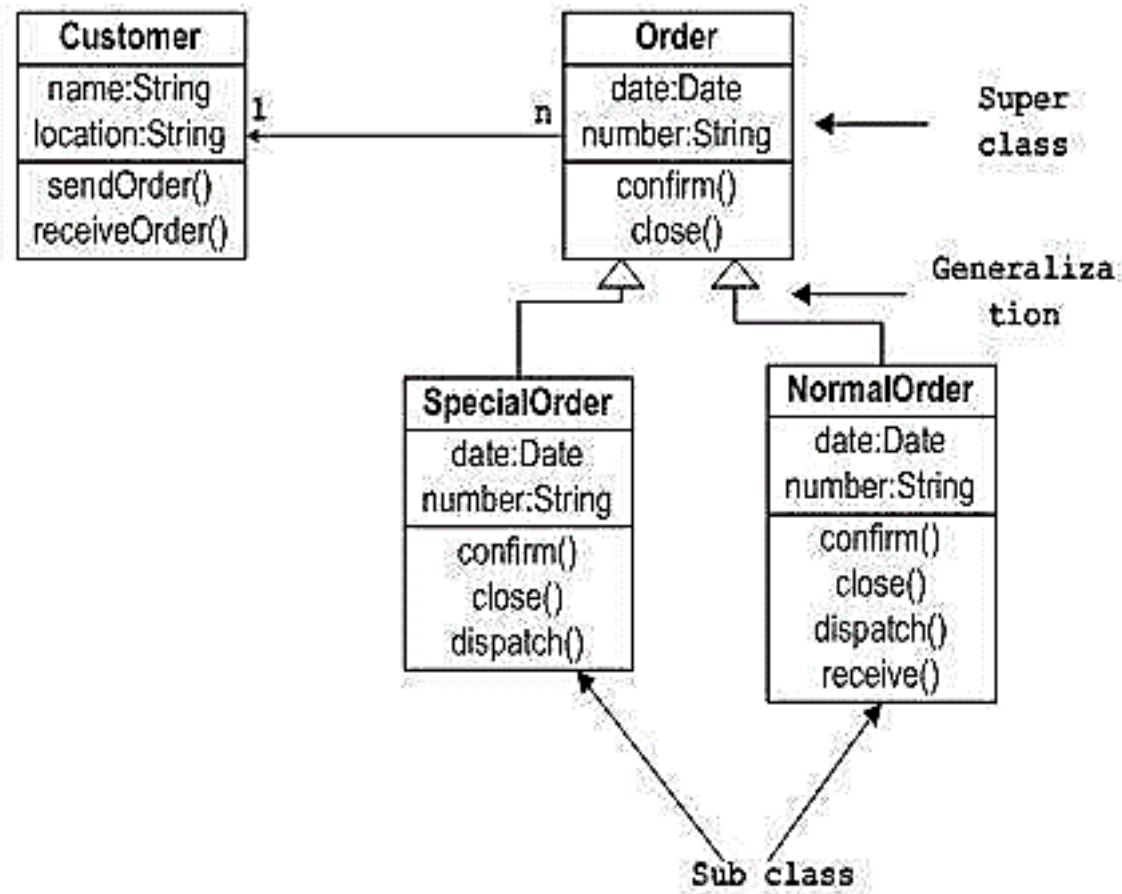
- Object Oriented Concepts
- Object Oriented Design with a Case Study



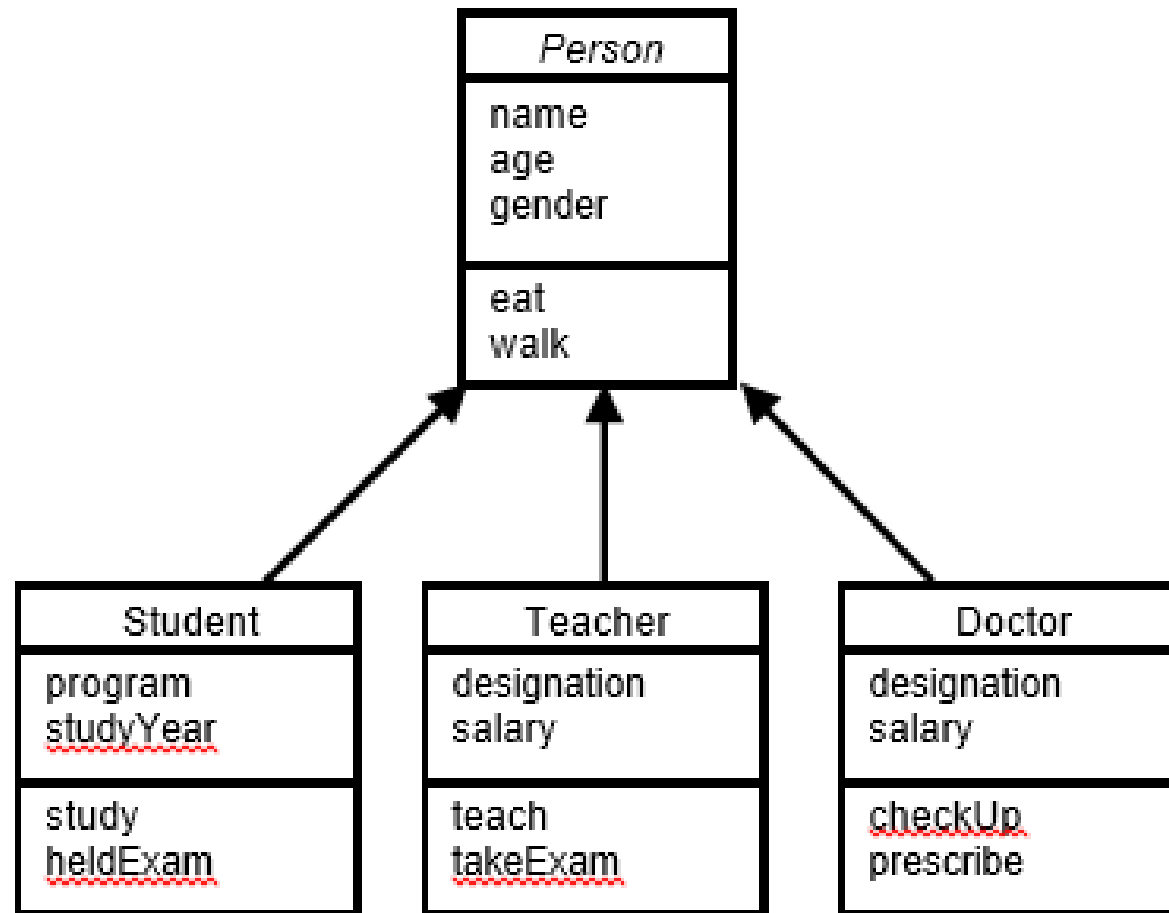
# OBJECT ORIENTED DESIGN CONCEPTS



# CLASS DIAGRAM



# INHERITANCE



## *Advantages of Inheritance*

- 1. Reuse*
- 2. Less redundancy*
- 3. Increased maintainability*

# KINDS OF ASSOCIATION

*There are two main types of association which are then further subdivided i.e.*

- Class Association
- Object Association

# CLASS ASSOCIATION

- *Class association is implemented in terms of Inheritance.*
- *Inheritance implements generalization/specialization relationship between objects.*
- *Inheritance is considered class association.*

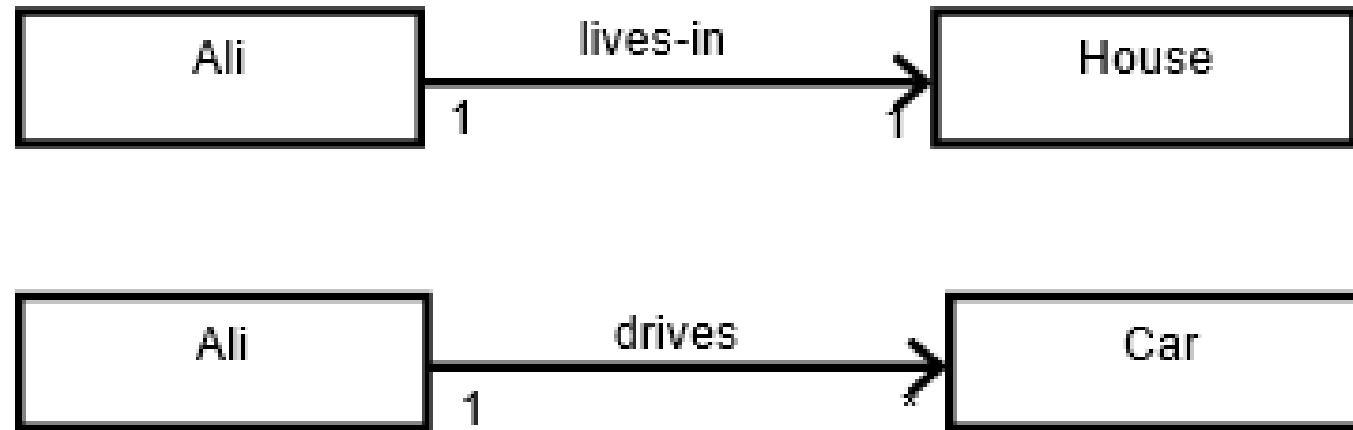
# OBJECT ASSOCIATION

It can be of one of the following types,

- *Simple Association*
- *Composition*
- *Aggregation*



# SIMPLE ASSOCIATION



It is generally called as “**association**” instead of “**simple association**”

# KINDS OF SIMPLE ASSOCIATION

Simple association can be categorized in two ways,

- With respect to direction (navigation)
- With respect to number of objects (cardinality)

## KINDS OF SIMPLE ASSOCIATION W.R.T NAVIGATION

With respect to navigation association has the following types,

- One-way Association
- Two-way Association

## ONE-WAY ASSOCIATION

In one-way association we can navigate along a single direction only, it is denoted by an arrow towards the server object.

### ■ Examples:



- Ali lives in a House



- Ali drives his Car

## TWO-WAY ASSOCIATION

In two-way association we can navigate in both directions, it is denoted by a line between the associated objects.



Employee works for company  
Company employs employees

## KINDS OF SIMPLE ASSOCIATION W.R.T CARDINALITY

With respect to cardinality association has the following types,

- Binary Association
- Ternary Association
- N-ary Association

## BINARY ASSOCIATION

It associates objects of exactly two classes; it is denoted by a line, or an arrow between the associated objects.

### Example



Association “works-for” associates objects of exactly two classes



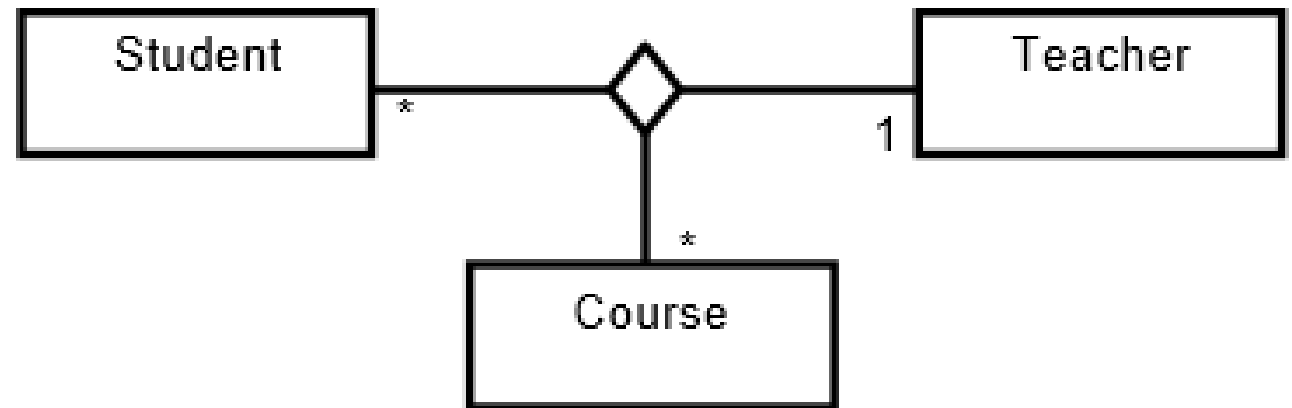
Association “drives” associates objects of exactly two classes

## TERNARY ASSOCIATION

It associates objects of exactly three classes; it is denoted by a diamond with lines connected to associated objects.

### **Example:**

Objects of exactly three classes are associated

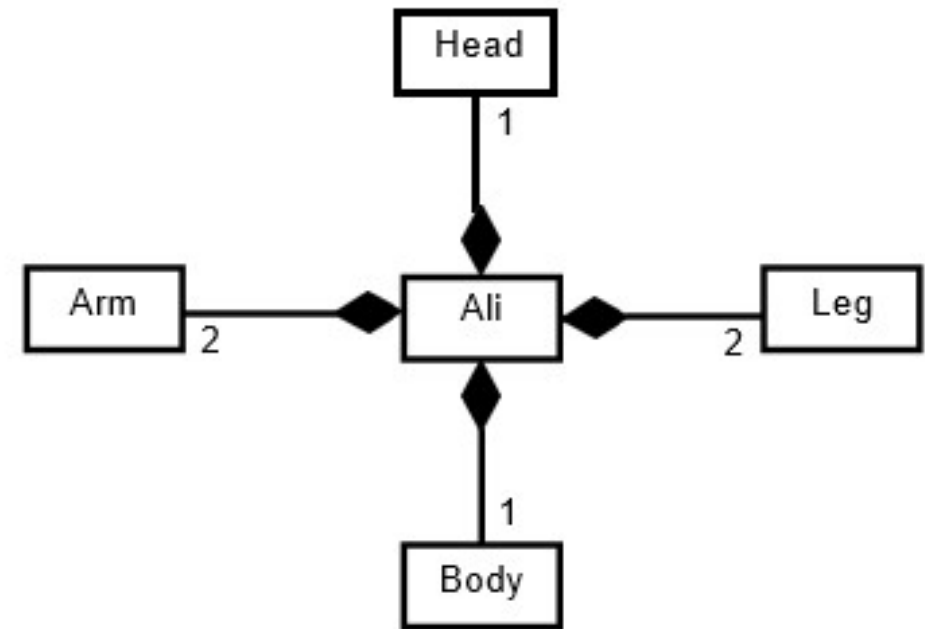




# COMPOSITION

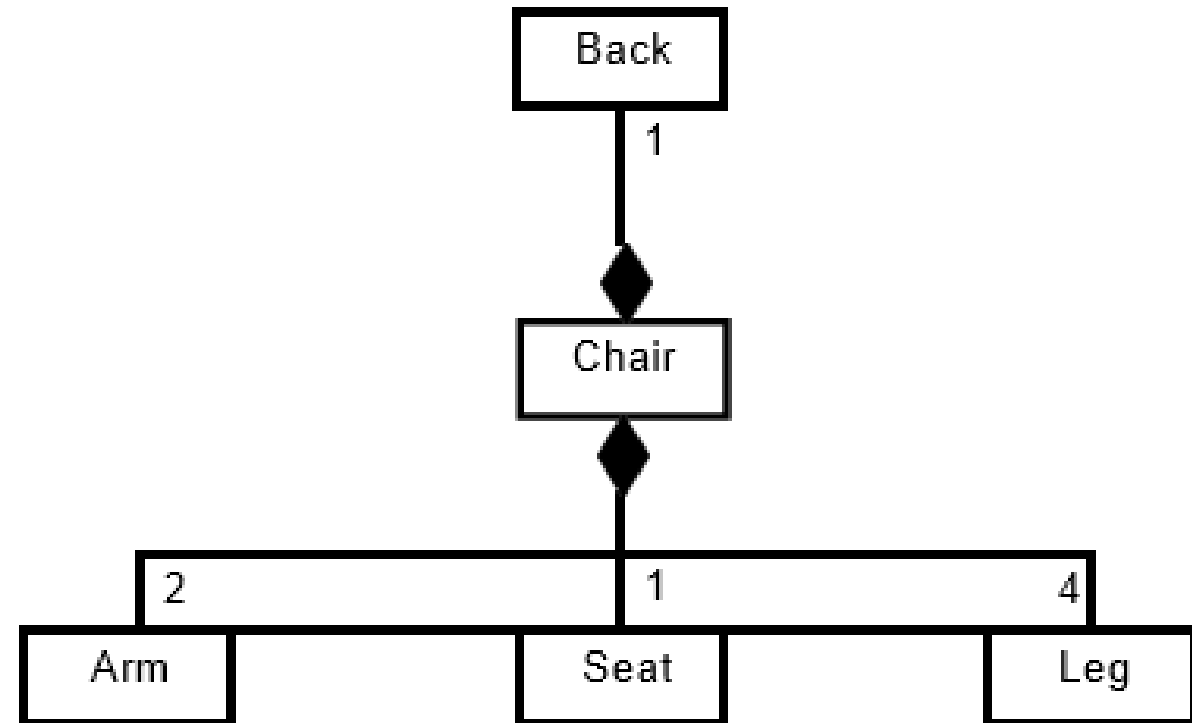
An object may be composed of other smaller objects, the relationship between the “part” objects and the “whole” object is known as Composition.

Composition is represented by a line with a filled-diamond head towards the composer object.



# COMPOSITION

Composition of chair:



# COMPOSITION

## **Composition is stronger relationship because**

- *Composed object becomes a part of the composer.*
- *Composed object can't exist independently.*

### **Example I**

- Ali is made up of different body parts They can't exist independent of Ali

### **Example II**

- Chair's body is made up of different parts They can't exist independently

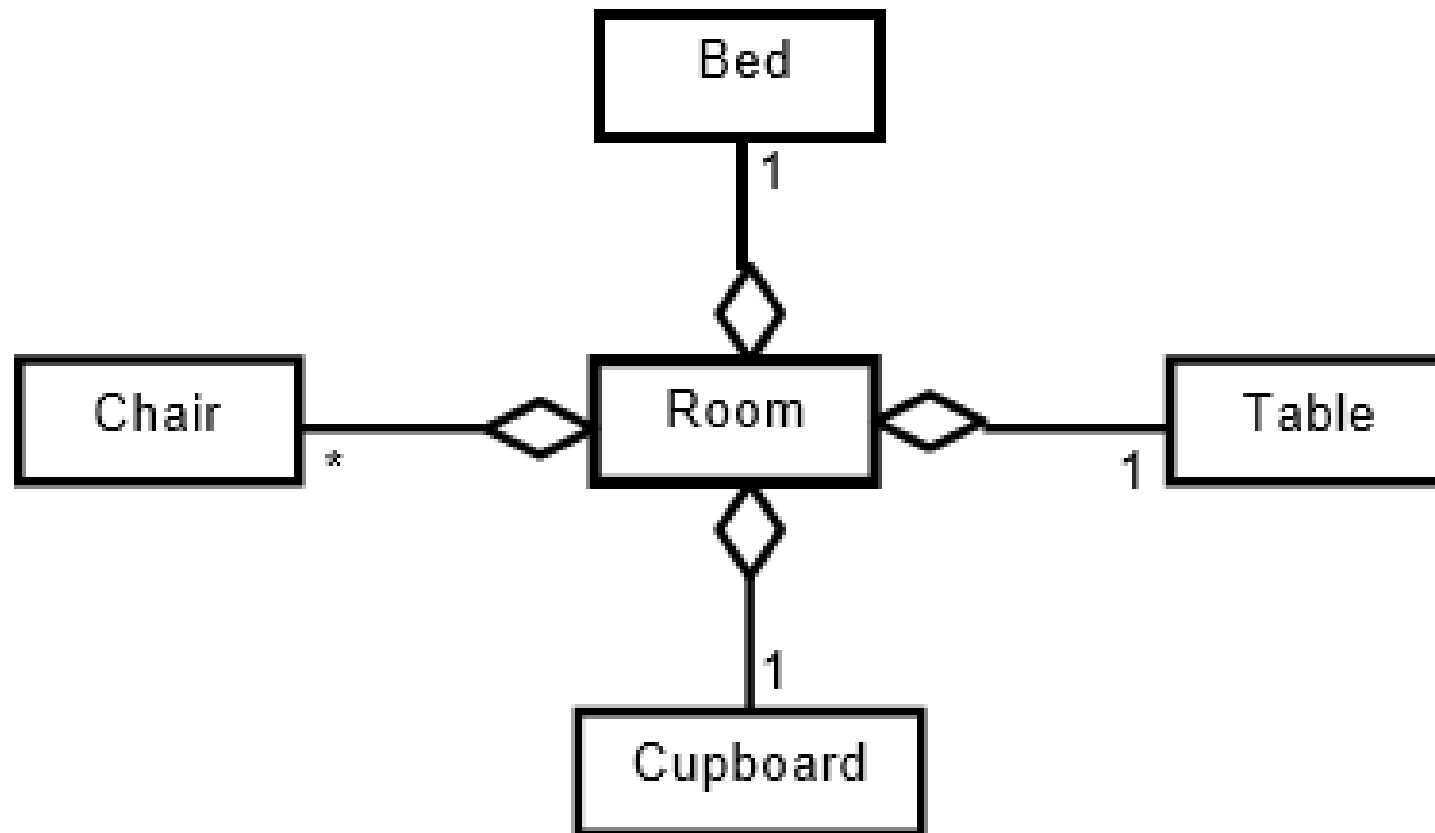
## AGGREGATION

- An object may contain a collection (aggregate) of other objects, the relationship between the container and the contained object is called aggregation,
- Aggregation is represented by a line with unfilled-diamond head towards the container.

### Example – Aggregation



## AGGREGATION



# AGGREGATION

## Aggregation is weaker relationship because

- *Aggregate object is not a part of the container*
- *Aggregate object can exist independently*

### Example I

- Furniture is not an intrinsic part of room
- Furniture can be shifted to another room, and so can exist independent of a particular room

### Example II

- A plant is not an intrinsic part of a garden
- It can be planted in some other garden, and so can exist independent of a particular garden

## POLYMORPHISM

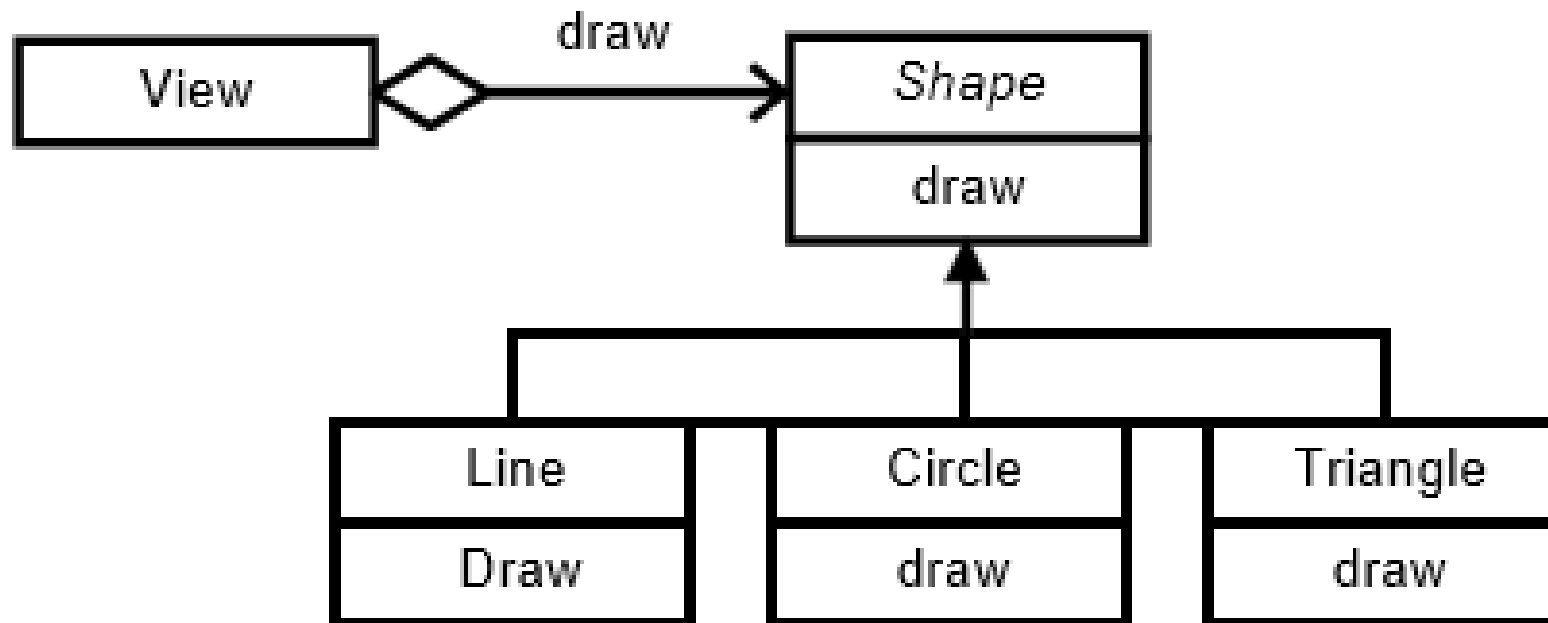
It is also essential component of object oriented modeling (paradigm).

In general, polymorphism refers to existence of ***different forms*** of a single entity.

For example, both Diamond and Coal are different forms of Carbon.

- *In OO model, polymorphism means that different objects can behave in different ways for the same message (stimulus).*

# POLYMORPHISM

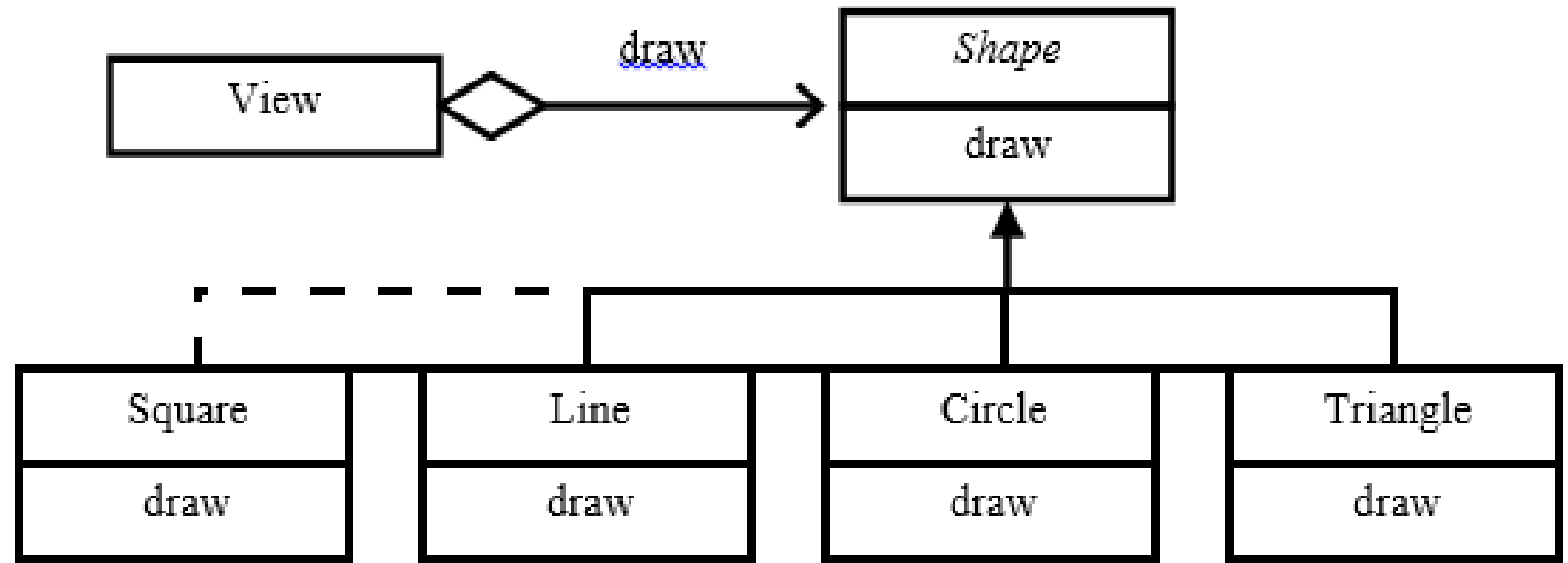




# POLYMORPHISM ADVANTAGES

Messages can be interpreted in different ways depending upon the receiver class.

New classes can be added without changing the existing model.

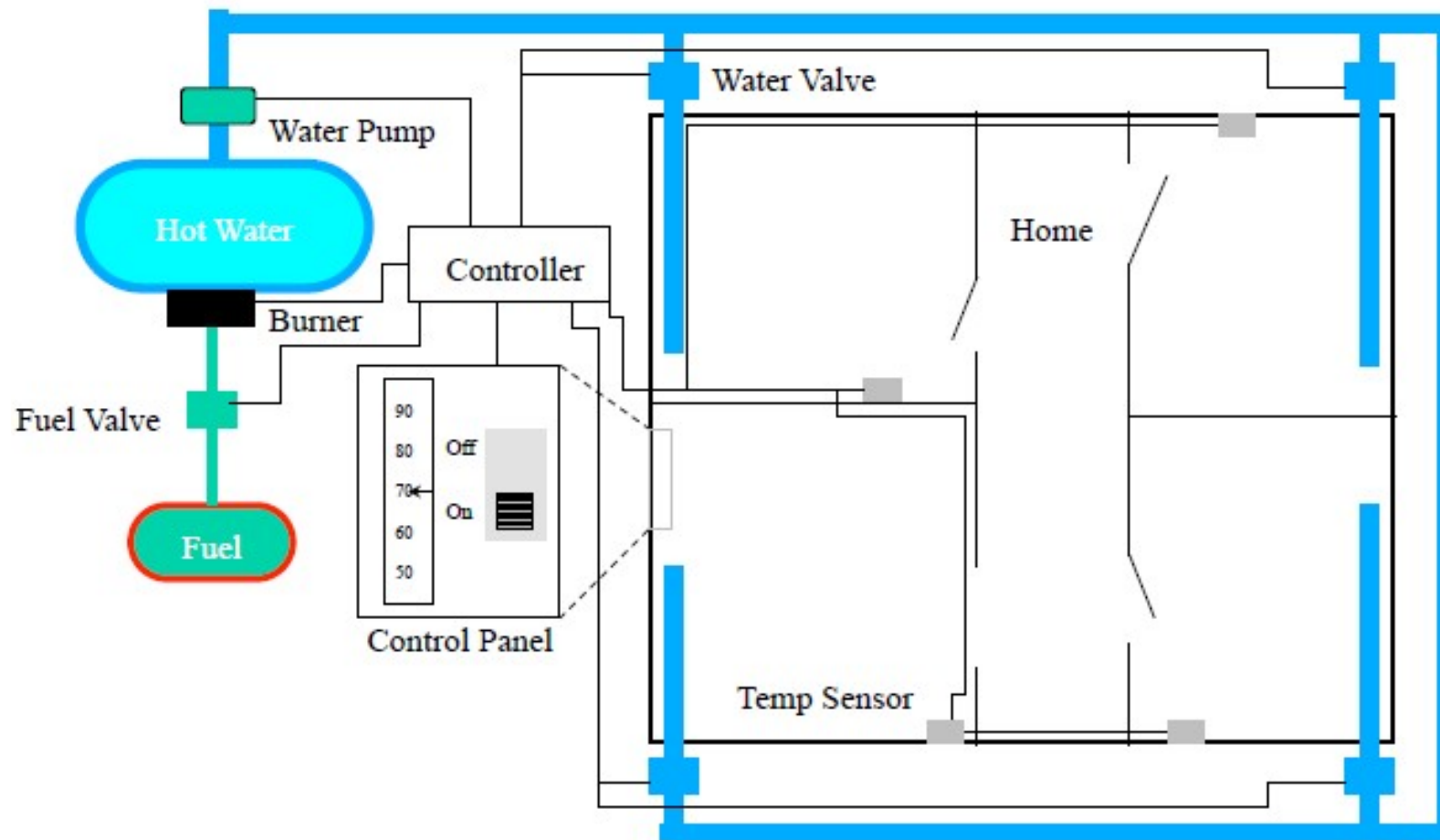


In general, polymorphism is a powerful tool to develop flexible and reusable systems



# OBJECT ORIENTED DESIGN WITH A CASE STUDY

# THE HOME HEATING SYSTEM



# HOME HEATING REQUIREMENTS

The purpose of the software for the Home Heating System is to control the heating system that heats the rooms of a house. The software shall maintain the temperature of each room within a specified range by controlling the heat flow to individual rooms.

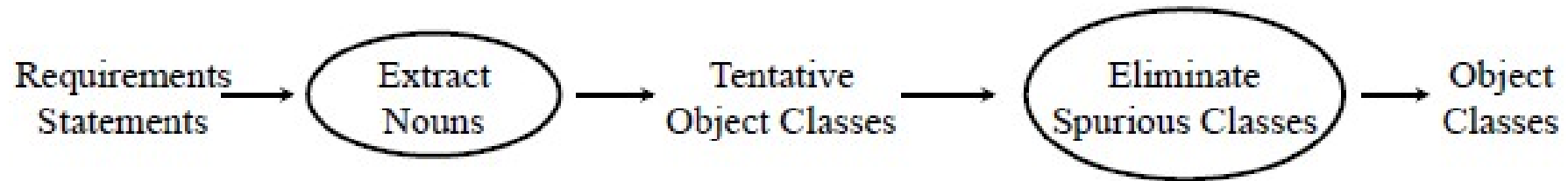
# HOME HEATING REQUIREMENTS

- The software shall control the heat in each room
- The room shall be heated when the temperature is 2F below desired temp
- The room shall no longer be heated when the temperature is 2F above desired temp
- The flow of heat to each room shall be individually controlled by opening and closing its water valve
- The valve shall be open when the room needs heat and closed otherwise
- The user shall set the desired temperature on the thermostat
- The operator shall be able to turn the heating system on and off

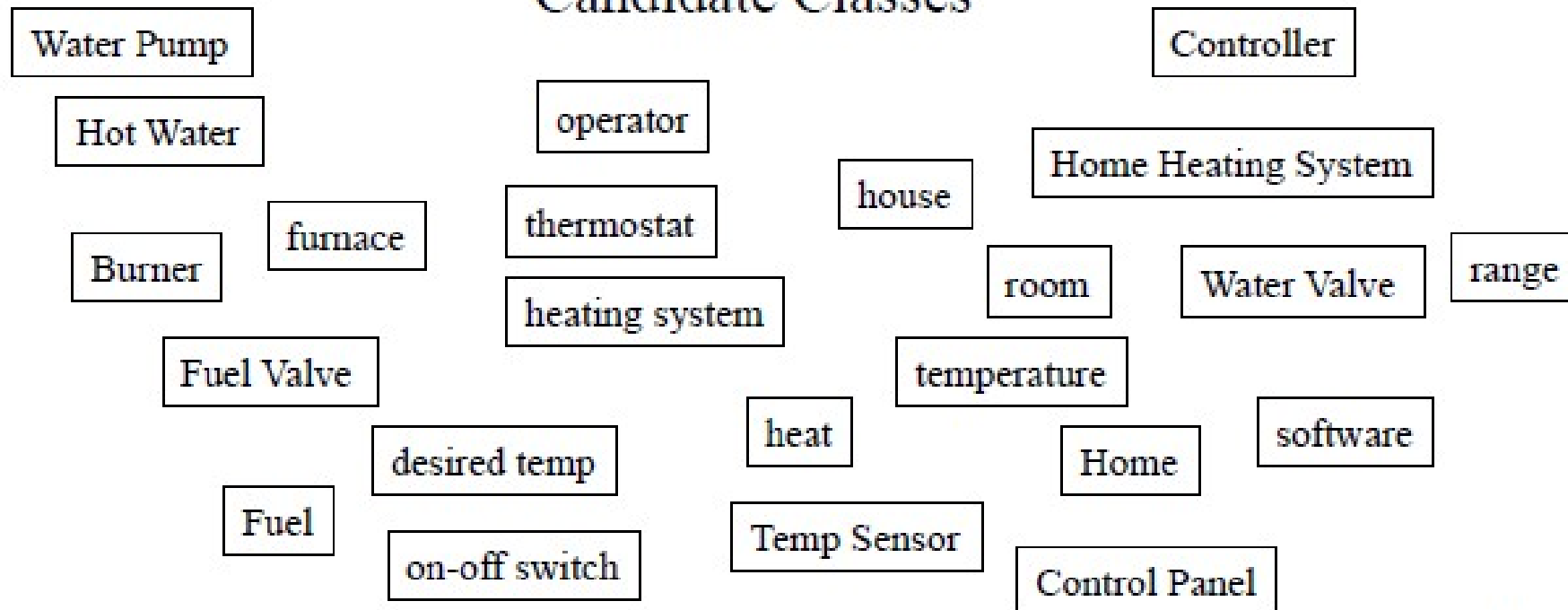
# HOME HEATING REQUIREMENTS

- The furnace must not run when the system is off
- When the furnace is not running and a room needs heat, the software shall turn the furnace on
- To turn the furnace on the software shall follow these steps
  - open the fuel valve
  - turn the burner on
- The software shall turn the furnace off when heat is no longer needed in any room
- To turn the furnace off the software shall follow these steps
  - close fuel valve

# IDENTIFY OBJECT CLASSES



## Candidate Classes



# ELIMINATION

Redundant

heating system

user

Irrelevant

Fuel

software

Hot Water

Vague

heat

house

heat flow

home

range

Attributes

desired temp

temperature

Operations

None

Roles

None

Implementation

None

Water Pump

Fuel Valve

thermostat

on-off switch

furnace

operator

Burner

room

Controller

Home Heating System

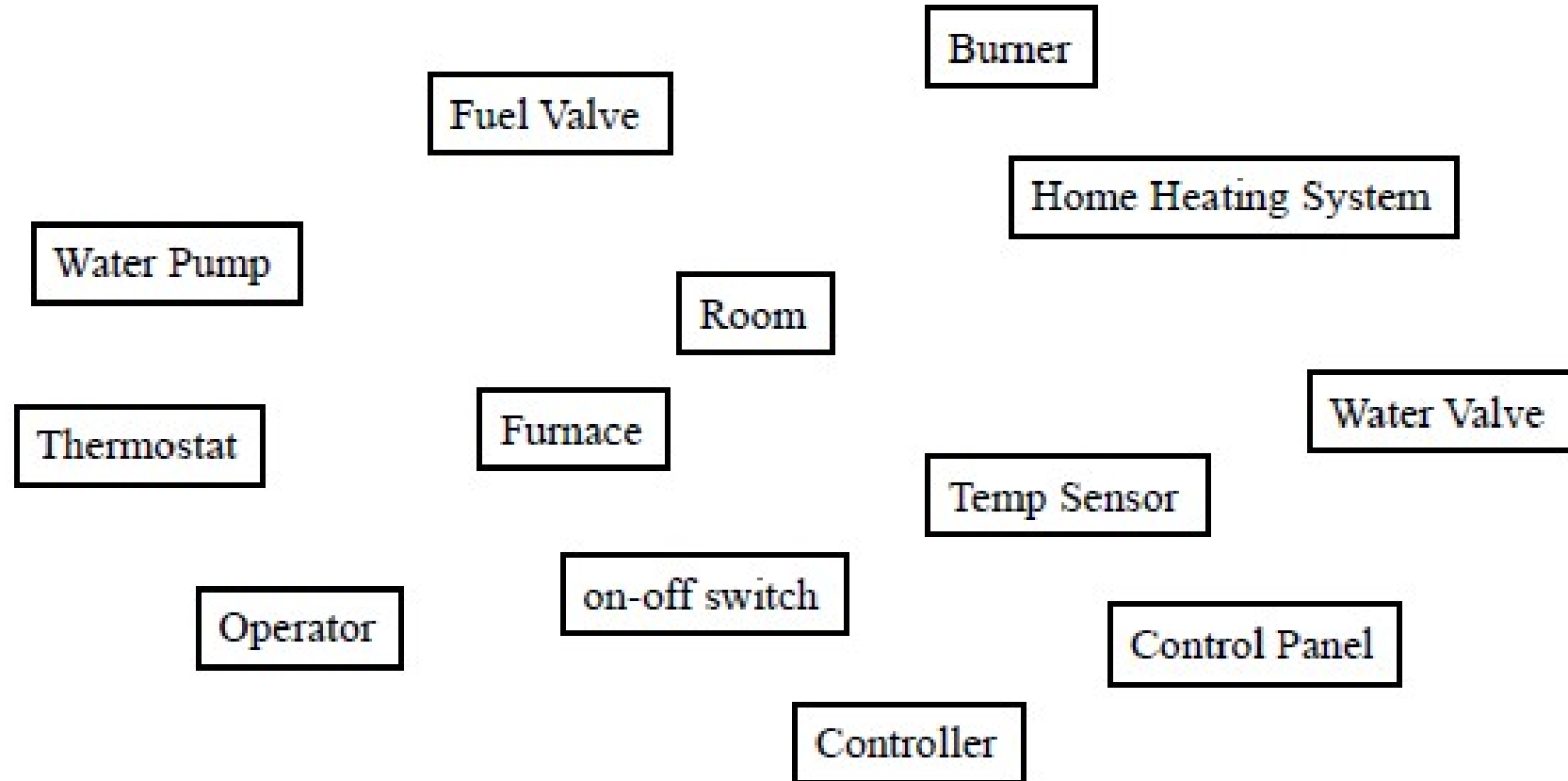
Temp Sensor

Control Panel

Water Valve



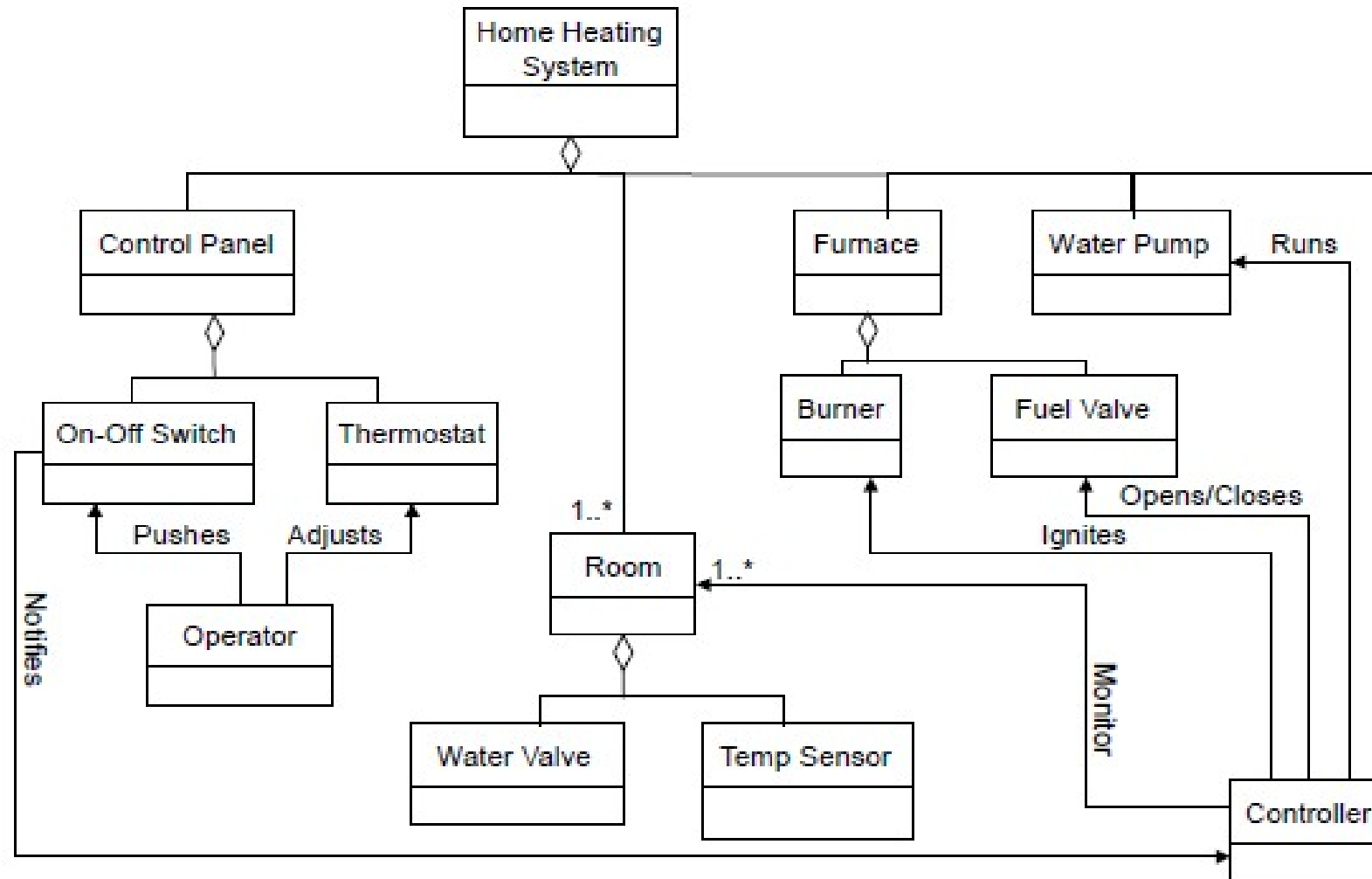
# AFTER ELIMINATION



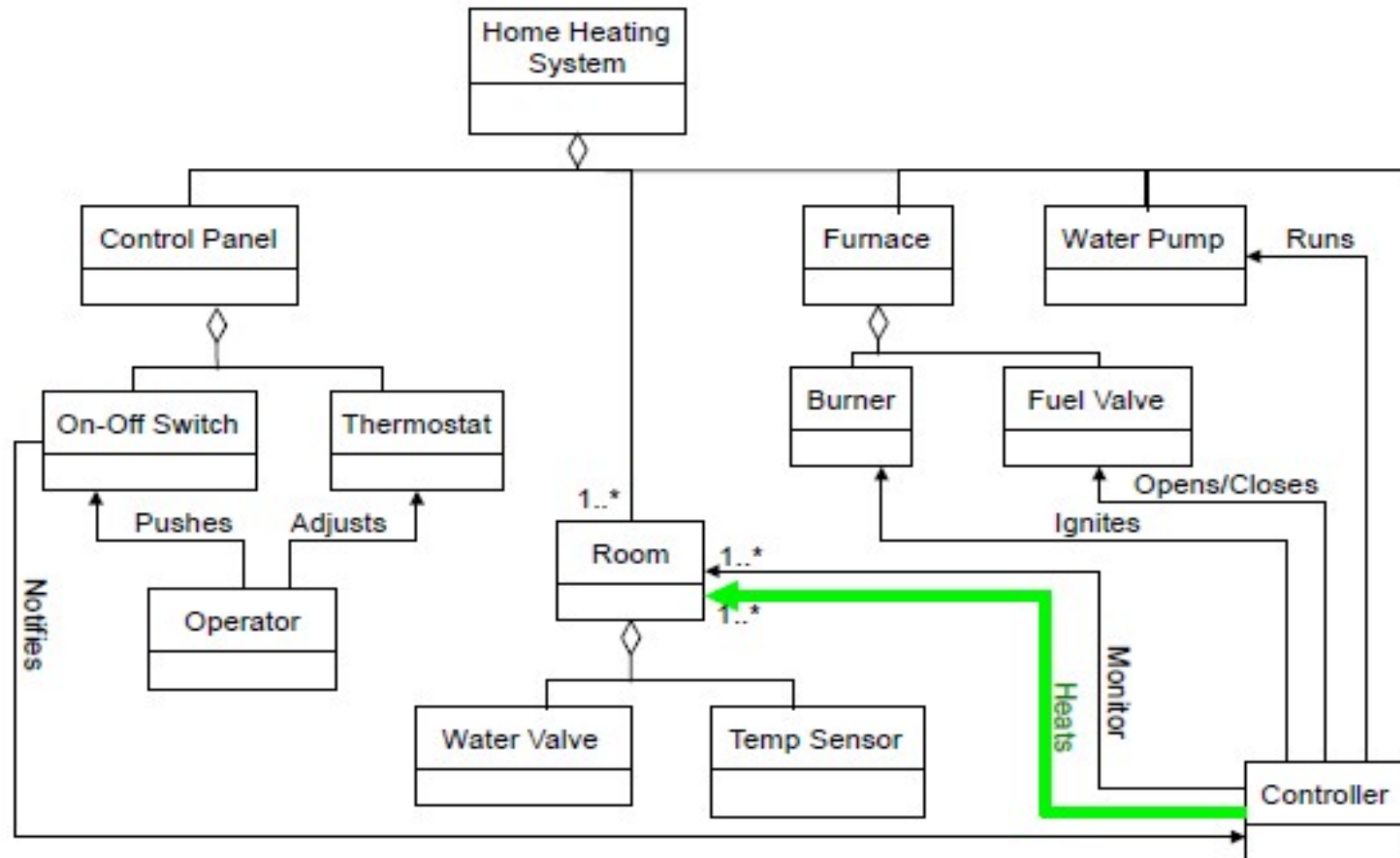
# POSSIBLE ASSOCIATIONS

- The home heating system **consists of** a furnace, rooms, a water pump, a control panel, and a controller
- The furnace **consists of** a fuel pump and a burner
- The control panel **consists of** an on-off switch and a thermostat
- The controller **controls** the fuel pump
- The controller **controls** the burner
- The controller **controls** the water pump
- The controller **monitors** the temperature in each room
- The controller **opens** and **closes** the valves in the rooms
- The operator **sets** the desired temperature
- The operator **turns** the system on and off

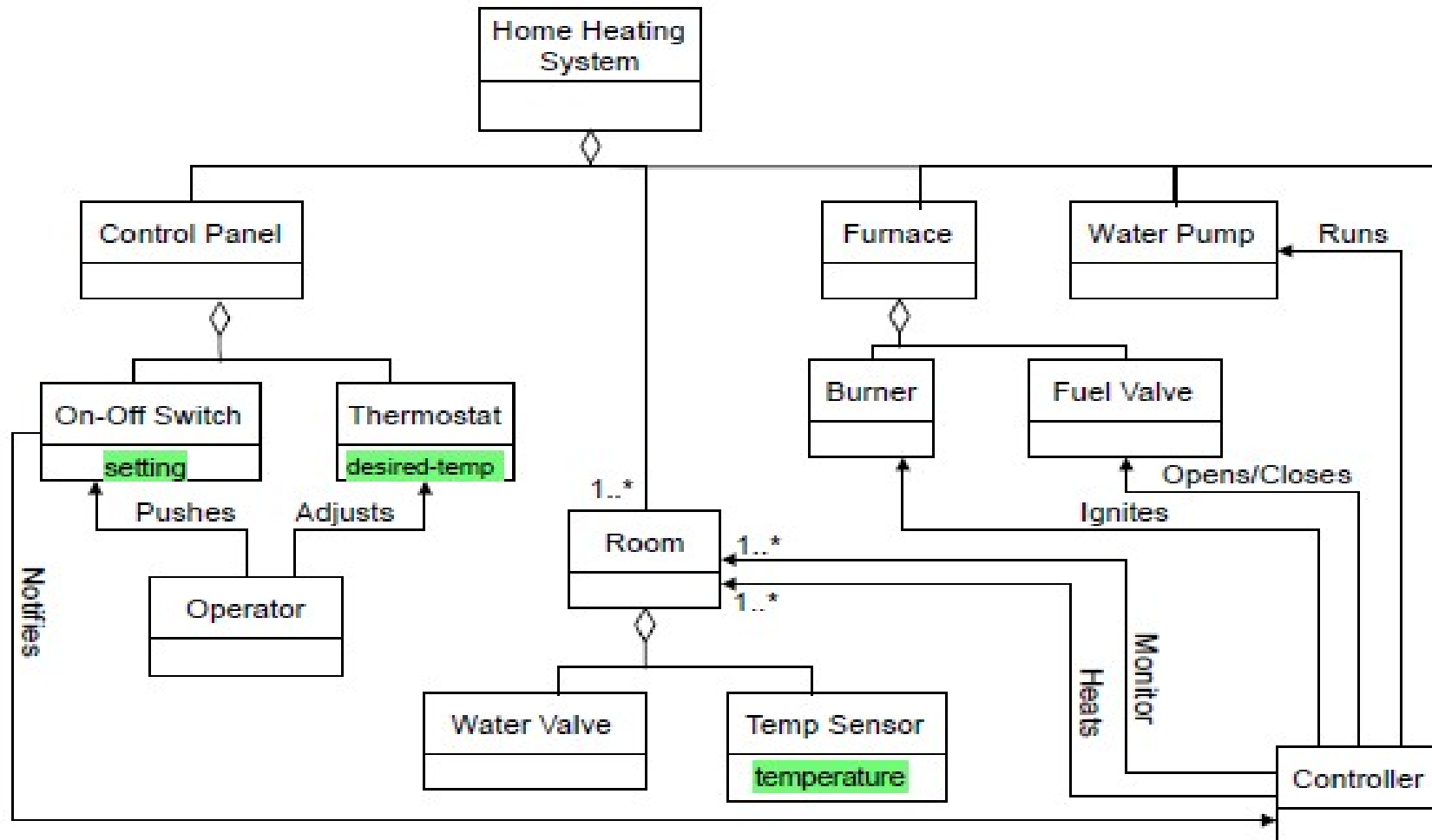
# OBJECT MODEL



# OBJECT MODEL - MODIFIED

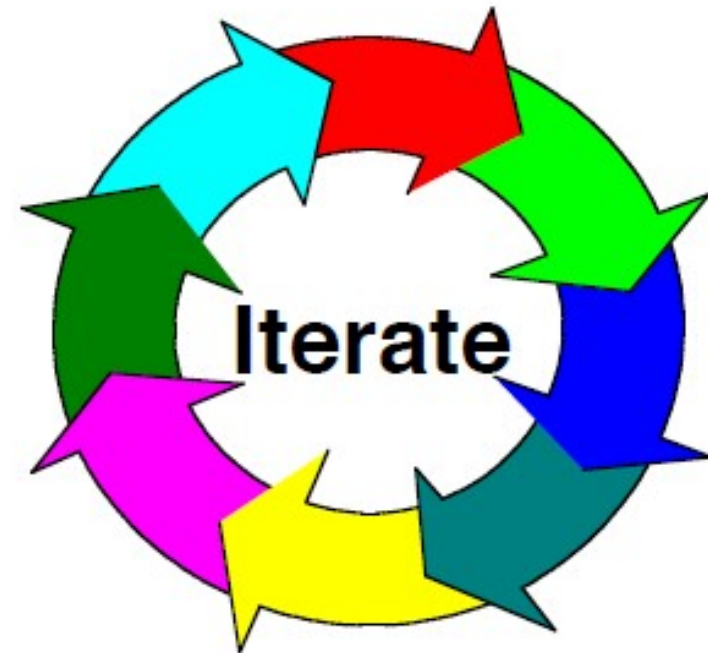


# OBJECT MODEL: ATTRIBUTES

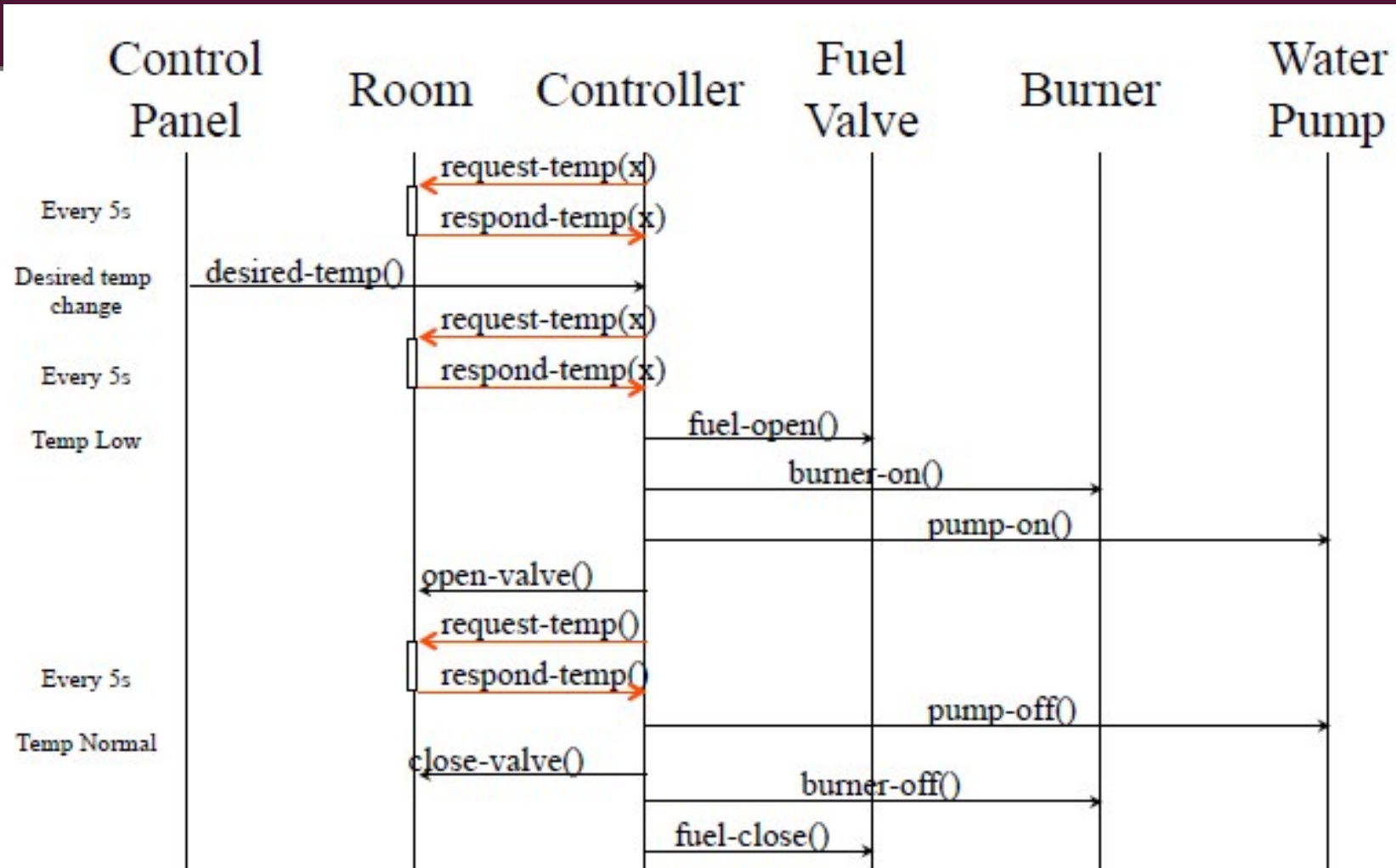


# ITERATE THE MODEL

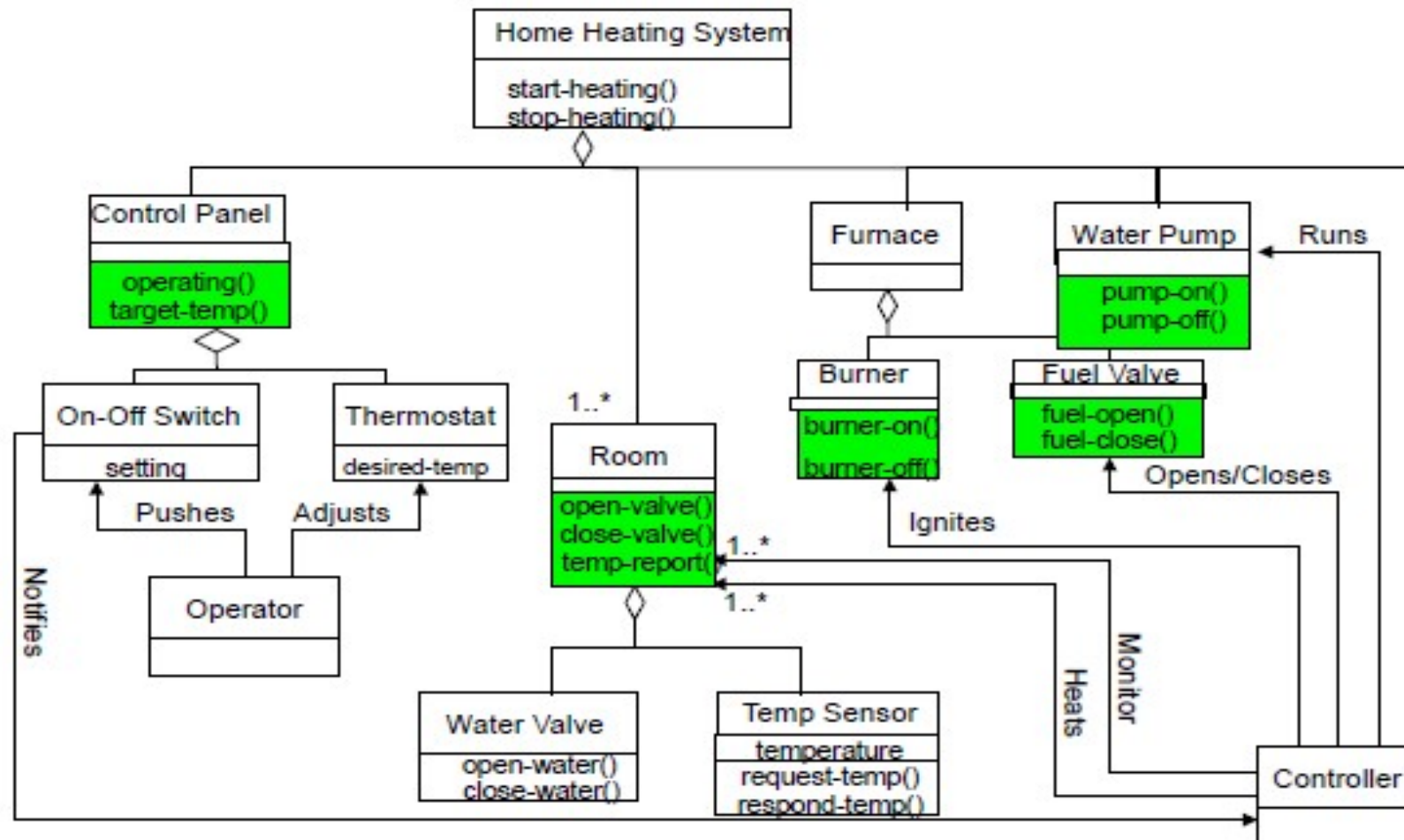
- Keep on doing this until you, your customer, and your engineers are happy with the model



# SEQUENCE DIAGRAM



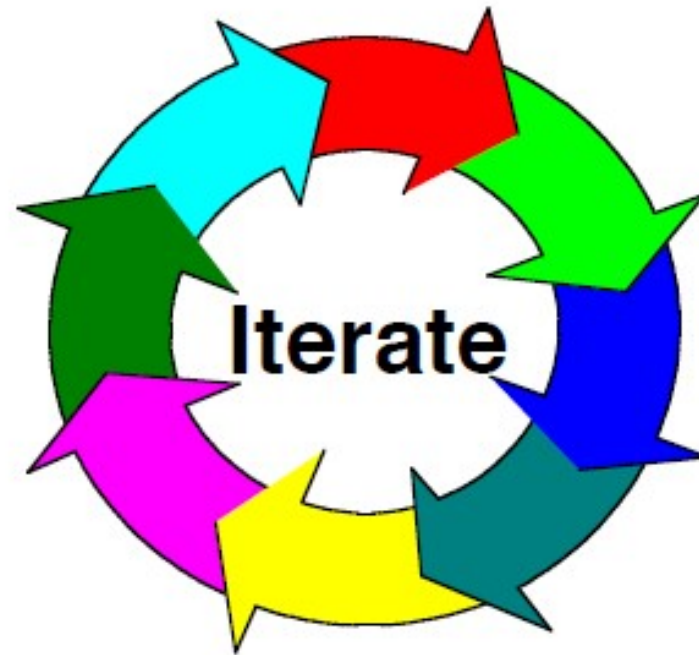
# OO MODEL – MODIFIED AGAIN!





# ITERATE THE MODEL

- Keep on doing this until you, your customer, and your engineers are happy with the model





**HAVE A GOOD DAY!**