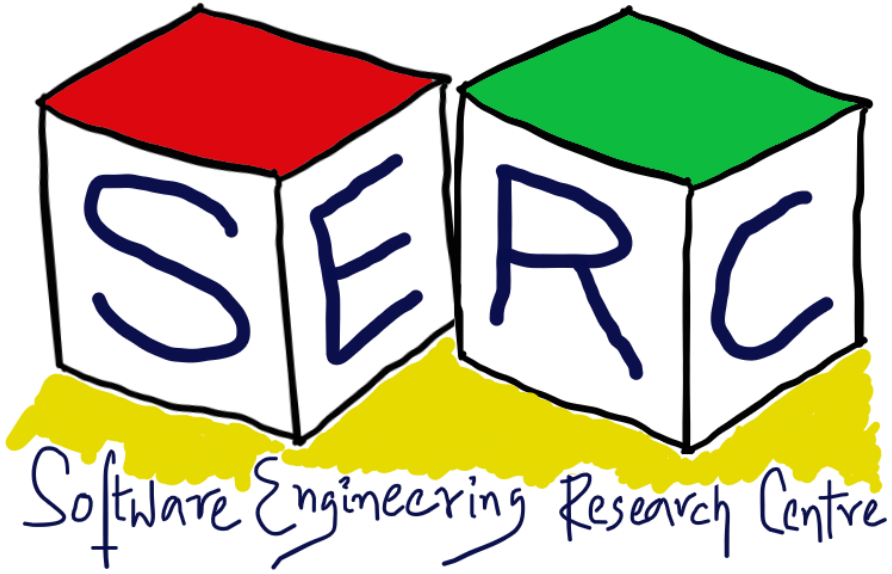
Software Modeling: An Overview

CS6.401 Software Engineering



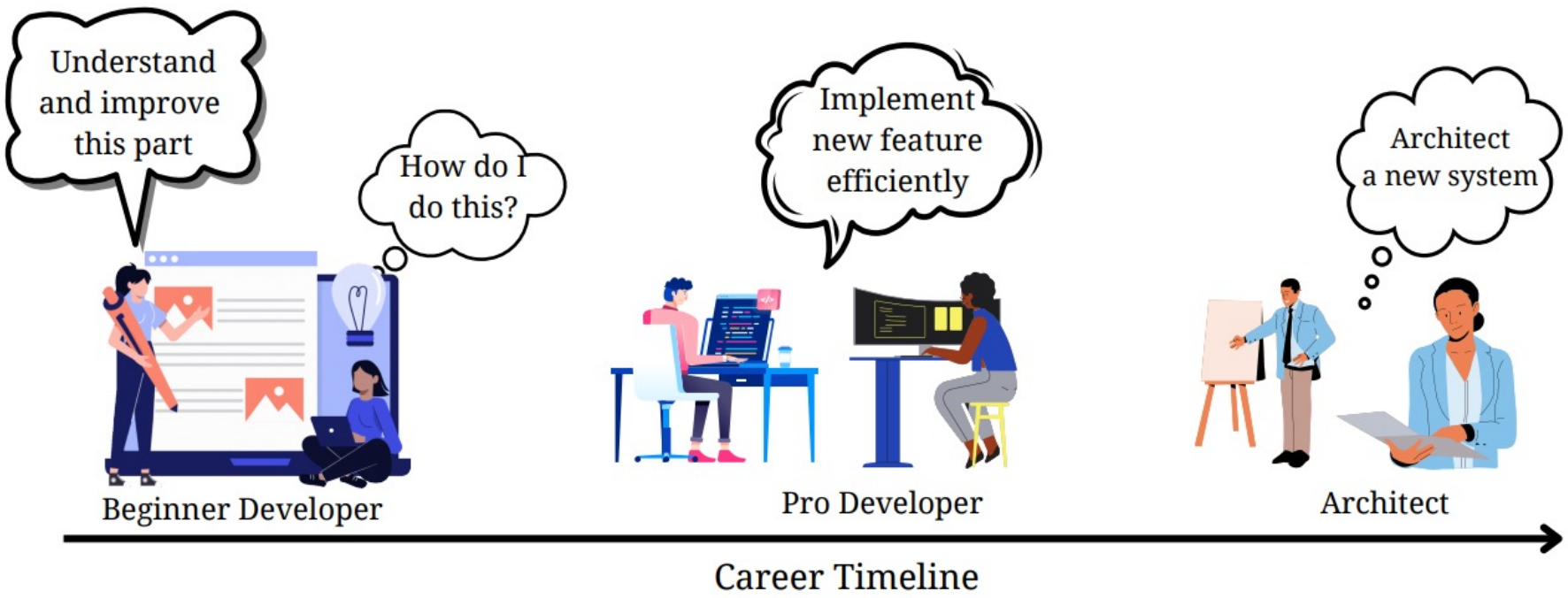
Dr. Karthik Vaidhyanathan

karthik.vaidhyanathan@iiit.ac.in

https://karthikvaidhyanathan.com

INTERNATIONAL INSTITUTE OF INFORMATION TECHNOLOGY
HYDERABAD

Course Outline



This Course

Modeling	Design Principles	Architectural Framework
Refactoring	Design Patterns	Architectural Patterns

Acknowledgements

The materials used in this presentation have been gathered/adapted/generate from various sources as well as based on my own experiences and knowledge

-- Karthik Vaidhyanathan

Sources:

1. Introduction to MDE, Ludovico Iovino, GSSI, Italy
2. UML@Classroom, An Introduction to Object-Oriented Modeling by Martina Seidl, Marion Scholz, Christian Huemer and Gerti Kappel
3. UML Modelling lecture, Dr. Raghu, IIIT Hyderabad

Source: Googe images

What is a Model?

Let us consider a real system

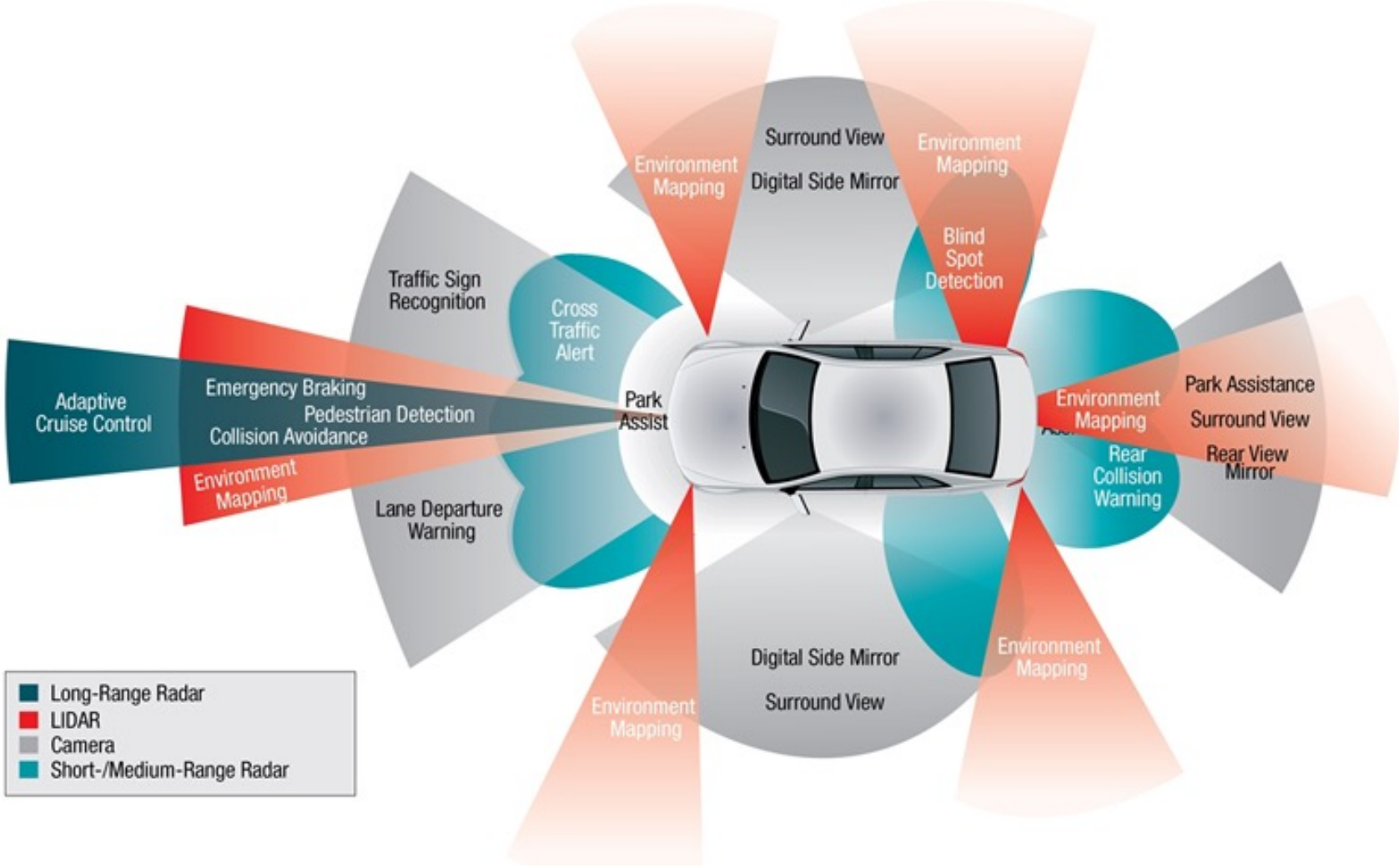
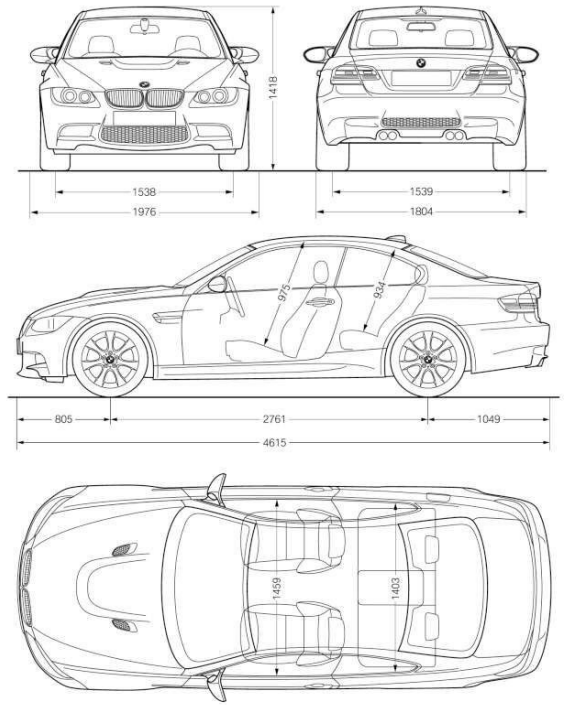


Image Source: machinedesign.com

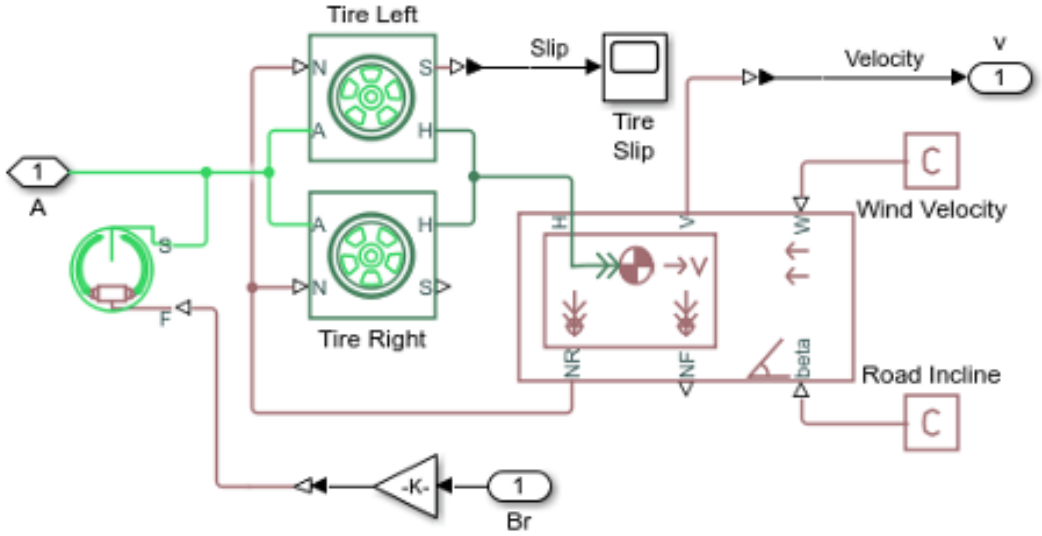
Let us now consider a model of the system

"*The brain does much more than recollect. It compares, synthesizes, analyzes, generates abstractions.*"

-- Carl Sagan



Model from the designers

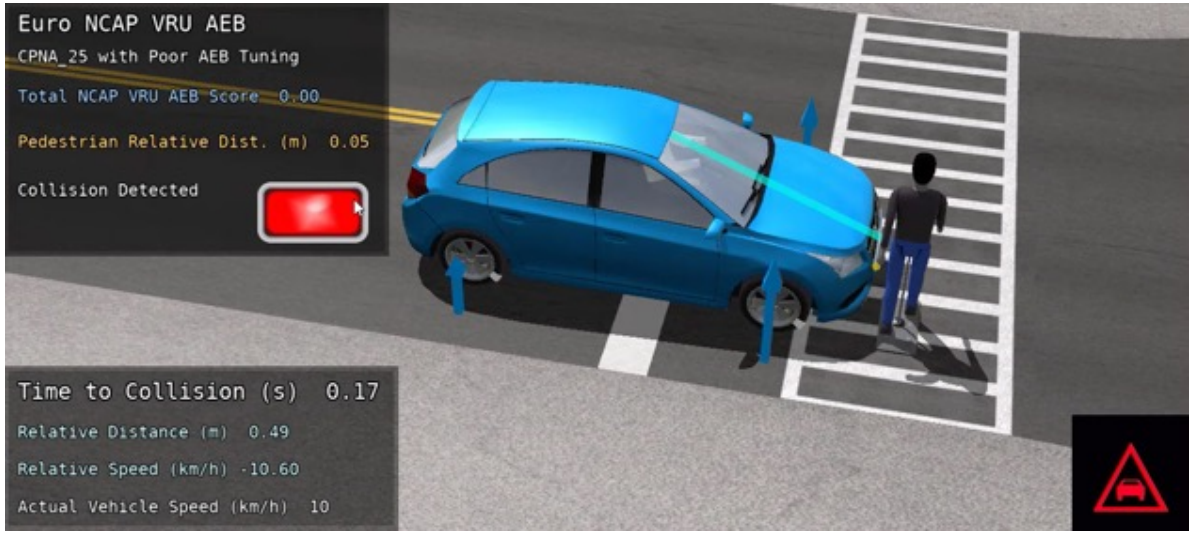


Model from the engineers

Model is a simplification of a reality. In other words, it is a blueprint of the system

Image source: Google images, mathworks

Modelling can serve more purpose



Checking collision avoidance

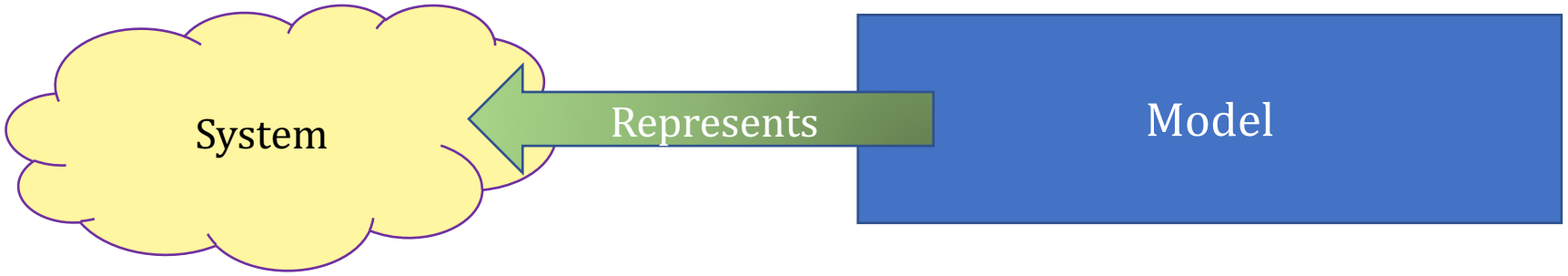
Checking if the traffic signs are followed

In essence models can be simulated with tools to perform analysis

Image source: carsim website

So what is a software model?

A simplified or partial representation of a real system, defined in order to **accomplish a task** or to **reach an agreement.**



Mapping: A model is always a mapping of some real system

Reduction: A model reflects only relevant set of properties of original system

Pragmatism: A model needs to be usable in place of the actual system with respect to some purpose

Quality of Model

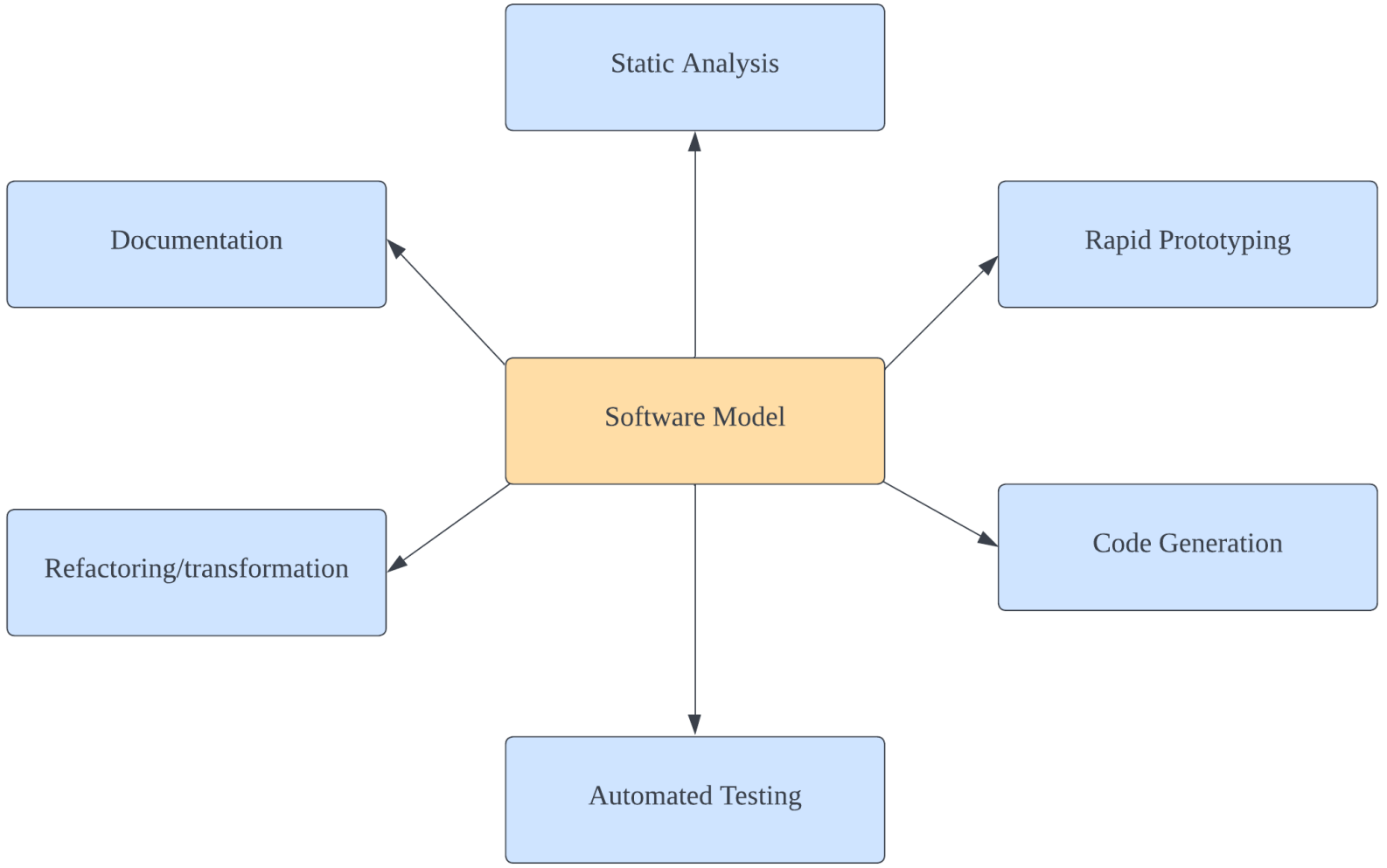
As per Bran Selic, five characteristics determine model's quality

- Abstraction: A model should be reduced version of system [Omit unwanted]
- Understandability: Should be as intuitive as possible
- Accuracy: Reflect relevant properties as close to reality as possible
- Predictiveness: Enable prediction of interesting properties of system
- Cost-effectiveness: Cheaper to create models than the system

Glimpse into world of Model-driven Engineering

Model Driven Software Engineering

Shifting focus from code centric techniques to models



What is MDE? – Key Motivation

Models as a sketch

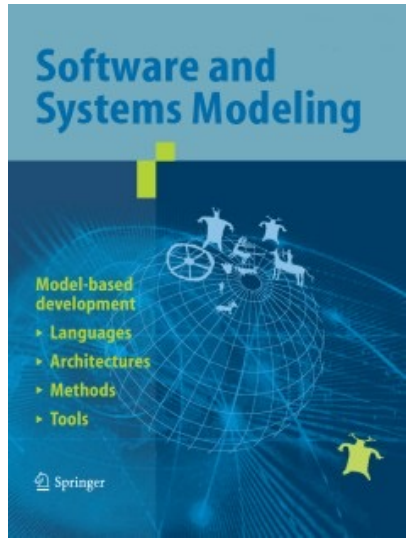
- Communication of ideas
- Objective: Modeling per se

Models as guidelines/blueprint

- Design decisions are documented
- Objective: instrumentation for implementation

Models as executable programs

- Generate code automatically
- Objective: models are source code and vice versa



Sosym journal

Models conference

Image source: models and sosym website

What to Model?

Multiple ways to think about it

Algorithmic Perspective

- Main block of building the software is procedure or function
- Scale and new features affects maintainability and reasoning

Object oriented Perspective

- Main building block of all software system is object or class
- Contemporary view of software development

Object Oriented Modeling

- Model system as a collection of objects that interact with each other

- Tries to captures the real-world scenario
  - Everything is an object
  - Has a state and behavior: (Happy, angry)…(speaking softly, yelling…)
  - Can you model a person?

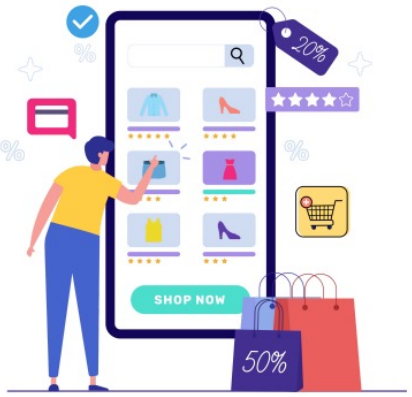
- Software objects are similar to real world objects:
  - Store state in fields (variables)
  - Behavior through methods (functions)

Objects vs Classes

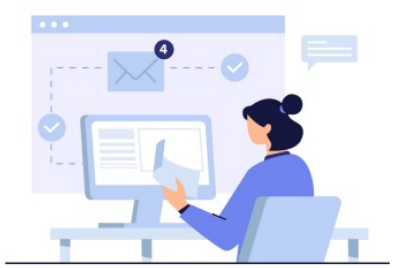
	Real world	Modeling World
Object	Object represents anything that can be distinctly identified	An object has identity, state and behavior
Class	Represents set of objects with similar characteristics and behavior	It characterizes the structure of states and behaviors shared by its instances.

Object is like a variable of a class

Lets take a scenario – E-commerce System, Lets think!



What users see



Organization/admins

Developers building system

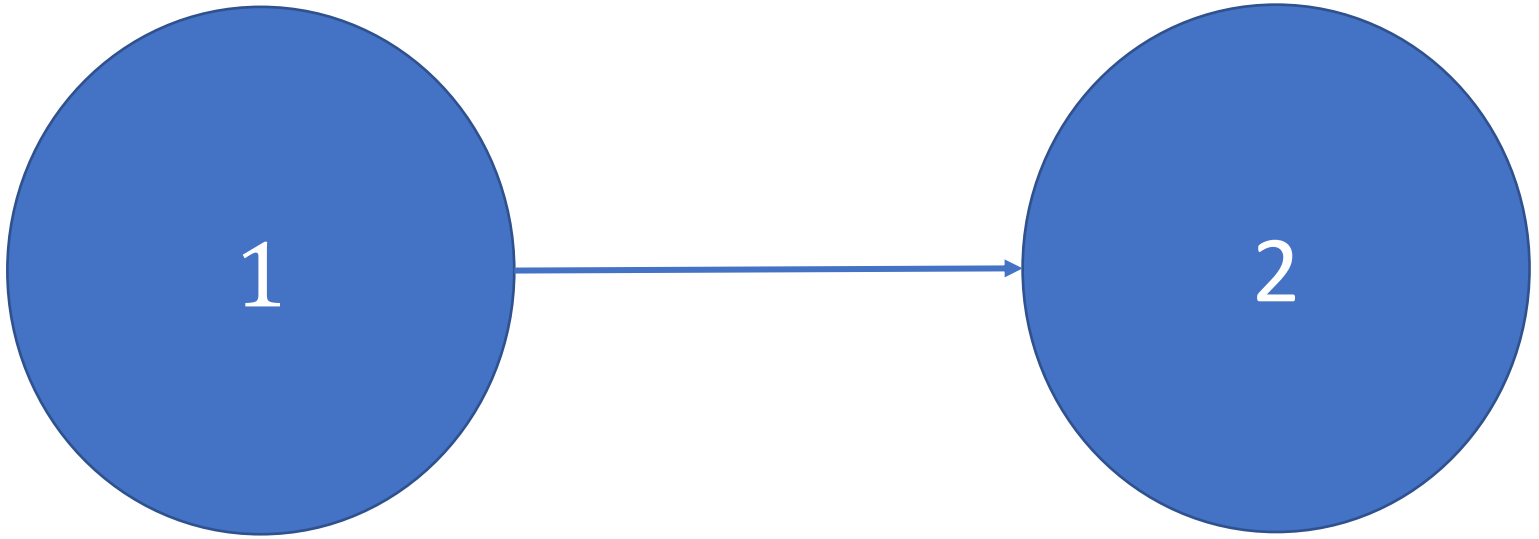
Behind the scenes!

Key Characteristics

- Abstraction: Hide irrelevant details (eg: Coffee machine)
  - Deals with What!
- Encapsulation: Protection against unauthorized access (eg: Organization)
  - Deals with how!
- Relationships
  - Inheritance: Derive classes from existing classes (eg: Real life inheritance!!)
  - Association: Relation between two classes (Aggregation, composition)
  - Dependency: Some form of dependency between two classes

How to Model?

How do you interpret this?



- 2 comes after 1
- 2 depends on 1
- 1 specializes/refines 2
- 2 listens to 1
- 1 contains 2

…

Can you create a model?

Think of a course management system like moodle. Can you create a model for the same?

Just use whatever knowledge you have, any type of diagram as per your knowledge is fine – Give a try!!

Thank You

Course website: karthikv1392.github.io/cs6401_se

Email: karthik.vaidhyanathan@iiit.ac.in
Web: https://karthikvaidhyanathan.com
Twitter: @karthi_ishere