TRINITY

TRINITY INSTITUTE OF TECHNOLOGY AND RESEARCH

RAJIV GANDHI PROUDYOGIKI VISHWAVIDYALAYA, BHOPAL

New Scheme Based On AICTE Flexible Curricula

Computer Science and Engineering, VII-Semester

CS701 Software Architectures

BY –ABHILASHA SAXENA

Unit 2

- Software architecture models: structural models, framework models, dynamic models, process models. Architectures styles: dataflow architecture, pipes and filters architecture, call-and return architecture, data-centered architecture, layered architecture, agent based architecture, Micro-services architecture, Reactive Architecture, Representational state transfer architecture etc.

Software architecture models

An **architectural model** (in **software**) is a rich and rigorous diagram, created using available standards, in which the primary concern is to illustrate a specific set of trade offs inherent in the structure and design.

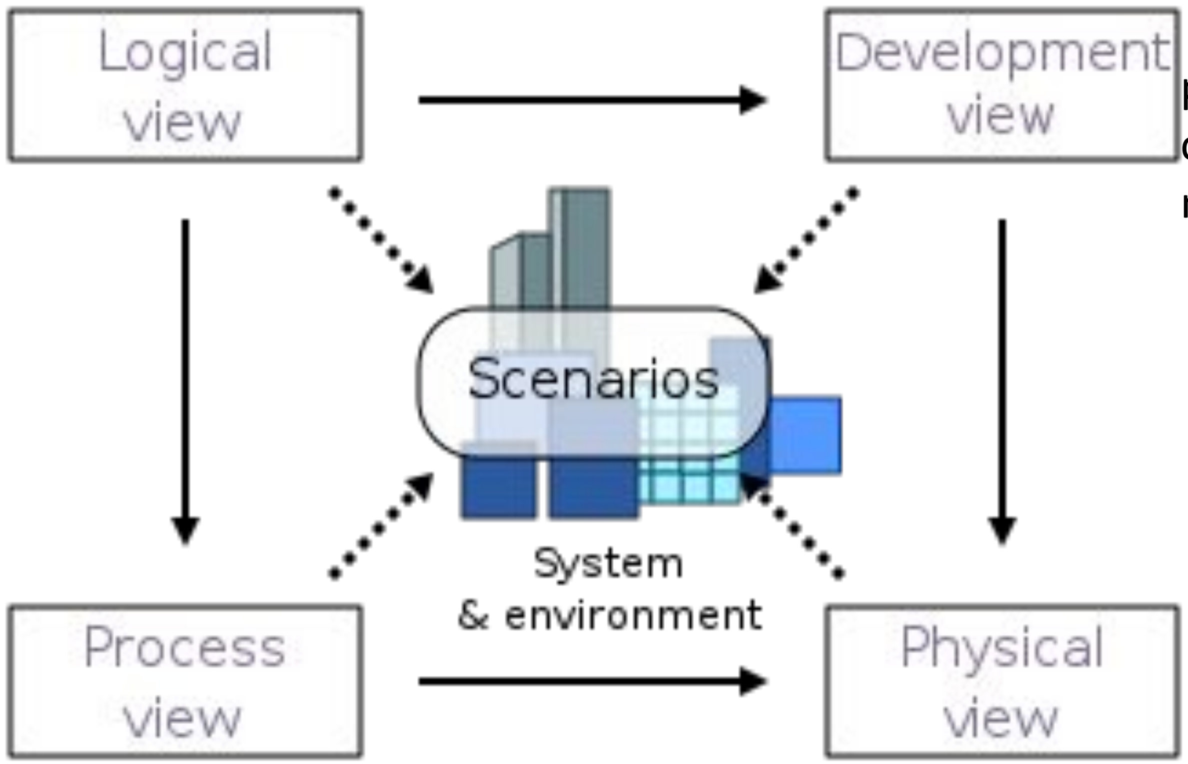
- What is a software architecture diagram?

The **Software Architecture Diagram** is a crucial step for **software** and application developers to describe the basic **software** structure by separating functional areas into layers. It depicts how a typical **software** system might interact with its users, external systems, data sources, and services.

Software architecture models

Concerned with functionality that the system provides to end users.

illustrates a system from a programmer's perspective and is concerned with software management.



Process view of work is defined as the understanding that work can be viewed as a "**process**" that has inputs, steps, and output(s) and that interfaces with other **processes** within an organization.

Physical view refers to the way data are physically stored and processed in a database.

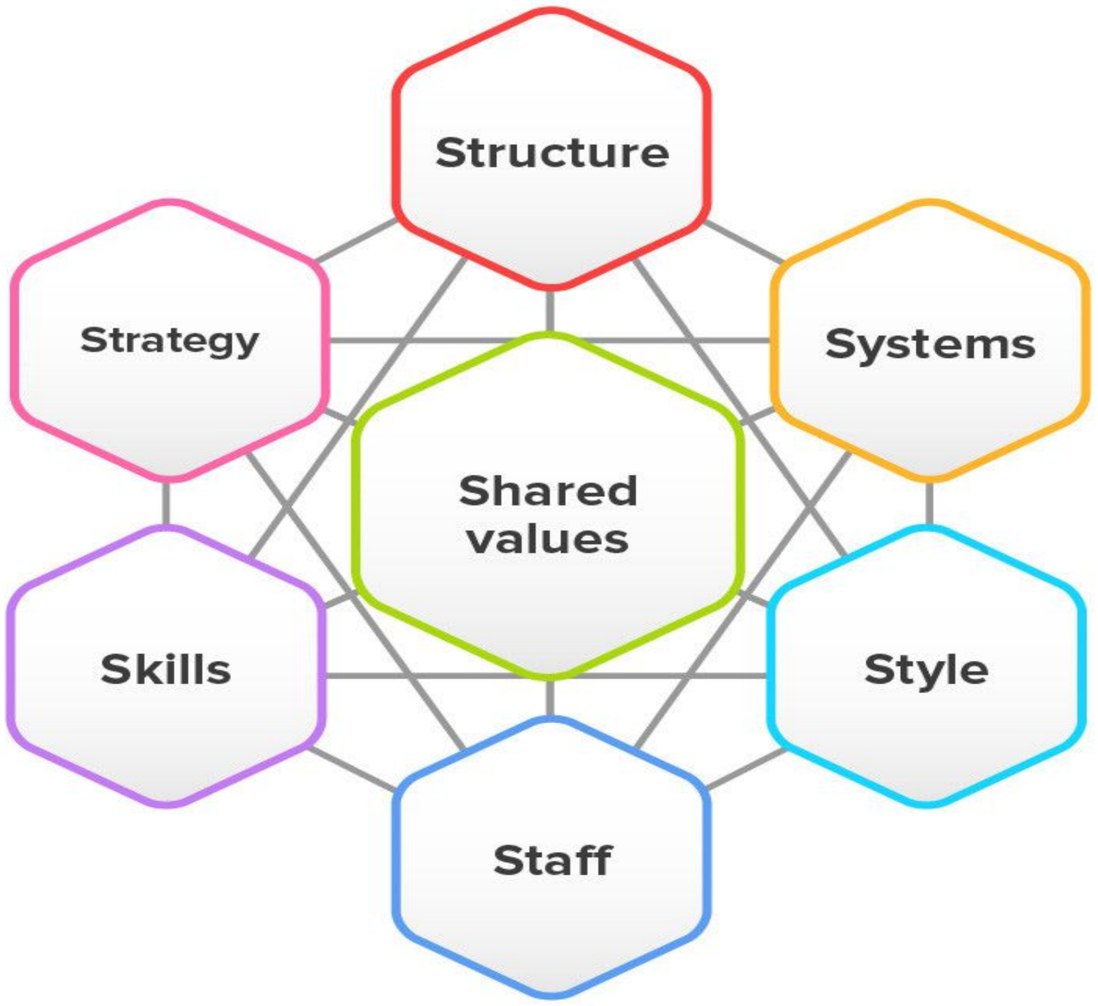
Software architecture

- Software architecture involves the high level structure of software system abstraction, by using decomposition and composition, with architectural style and quality attributes. A software architecture design must conform to the major functionality and performance requirements of the system, as well as satisfy the non-functional requirements such as reliability, scalability, portability, and availability.

Structural models

- Structural model of software display the organization of a system in terms of the components that make up that system and their relationship.
- It may be static or dynamic.
- Static model which shows the structure of the system design and dynamic models which shows the organization of the system when it is executing.

- What does a framework mean?
- A **framework**, or software **framework**, is a platform for developing software applications. It provides a foundation on which software developers can build programs for a specific platform. ...
  A **framework** may also include code libraries, a compiler, and other programs used in the software development process.
- A **model** is the presentation in schematic form, often in a simplified way, of an existing or future state or situation. ... A **framework** is an entity between a '**model**' and a 'method'. A **framework** is, or contains, a (not completely detailed) structure or system for the realization of a defined result/goal.


Structure
Strategy
Systems
Shared values
Skills
Style
Staff

Dynamic models

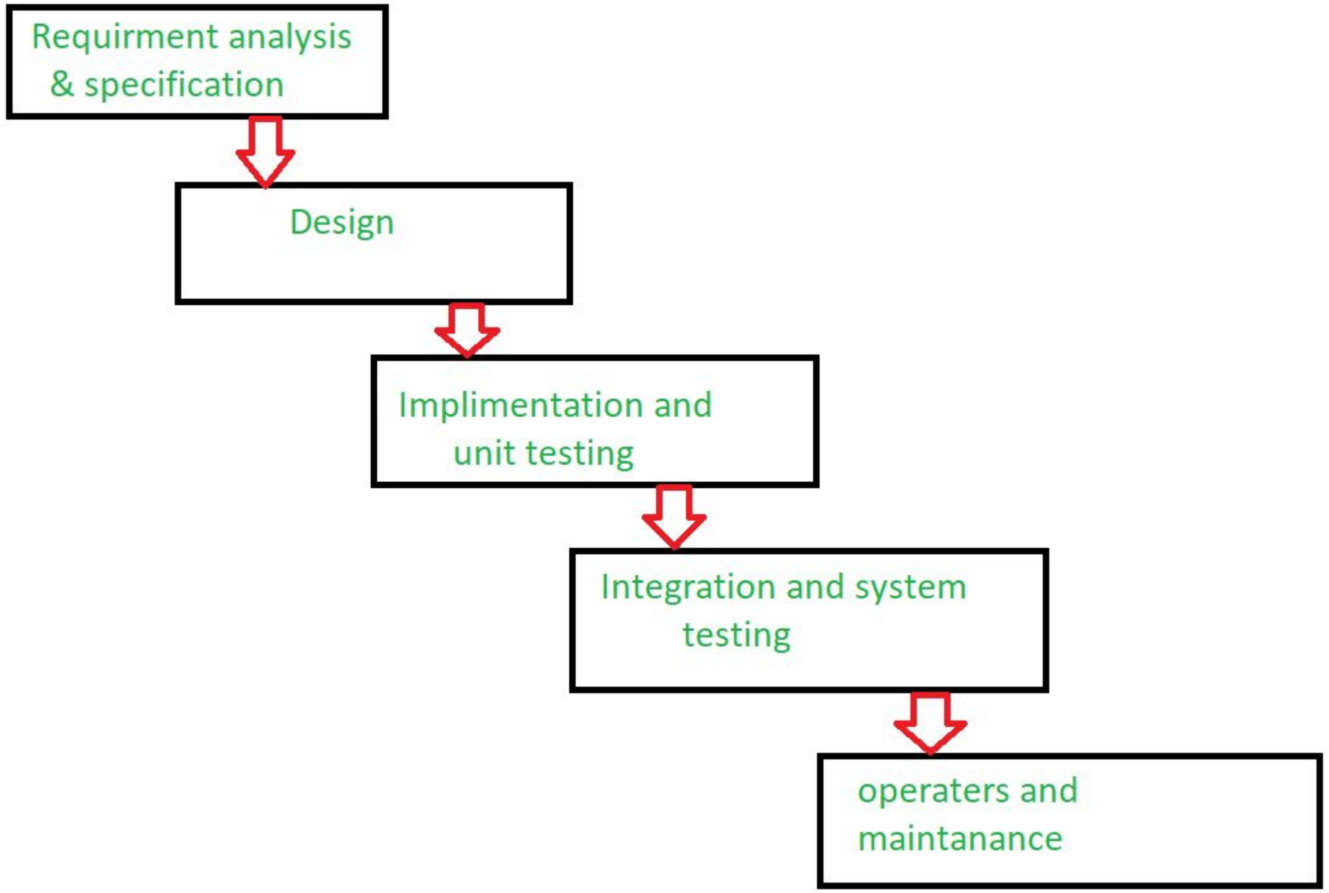
- The **dynamic model** is used to express and **model** the behaviour of the system over time. It includes support for activity diagrams, state diagrams, sequence diagrams and extensions including business process modelling.

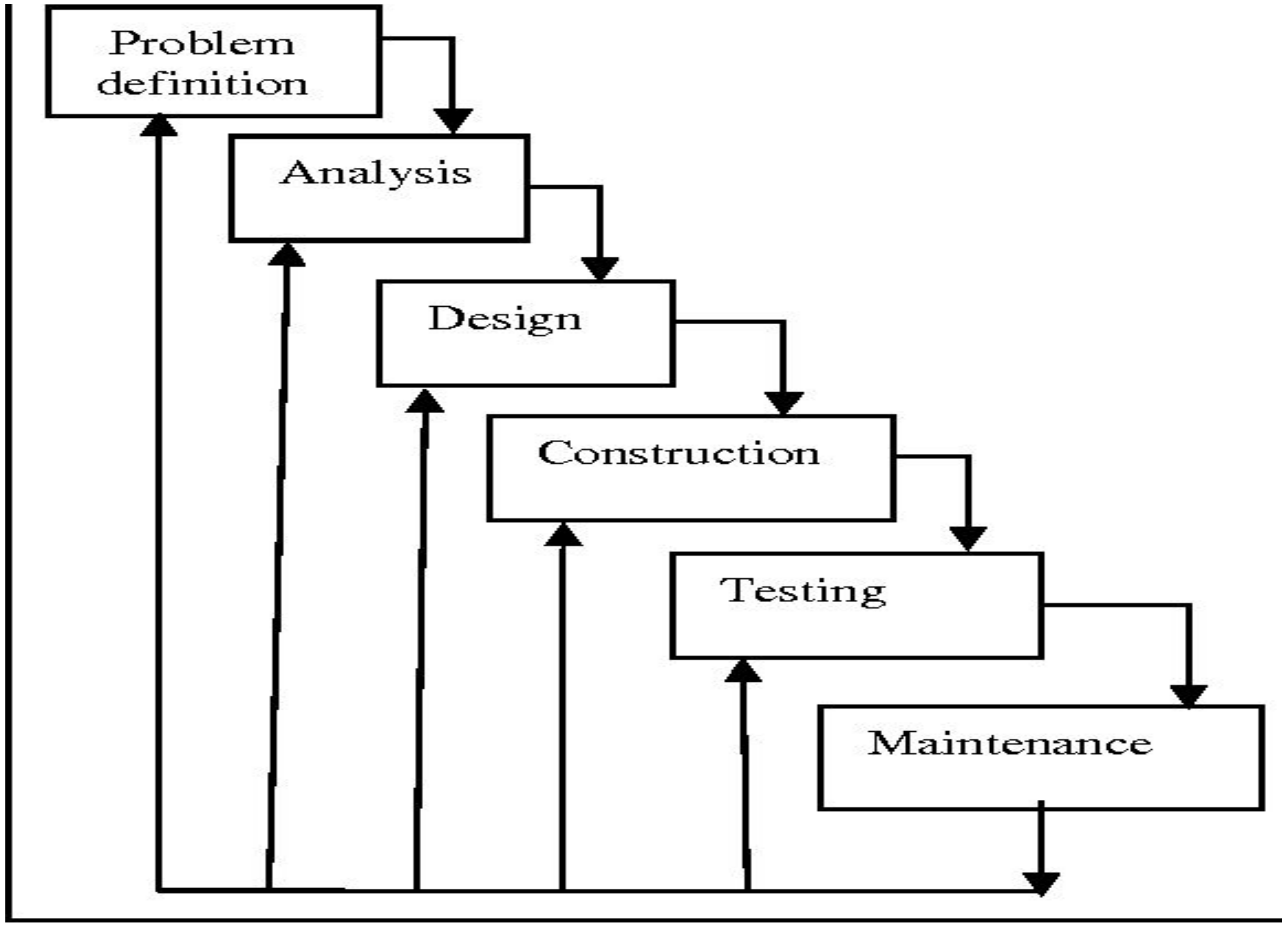
Static vs. dynamic models

- Static model
  - Describes the static structure of a system
  - Consists of a set of objects (classes) and their relationships
  - Represented as class diagrams
- Dynamic model
  - Describes the dynamic behavior of a system, such as its state transitions and interactions (message sends)
  - Represented as statechart, activity, sequence, and communication diagrams

Process models

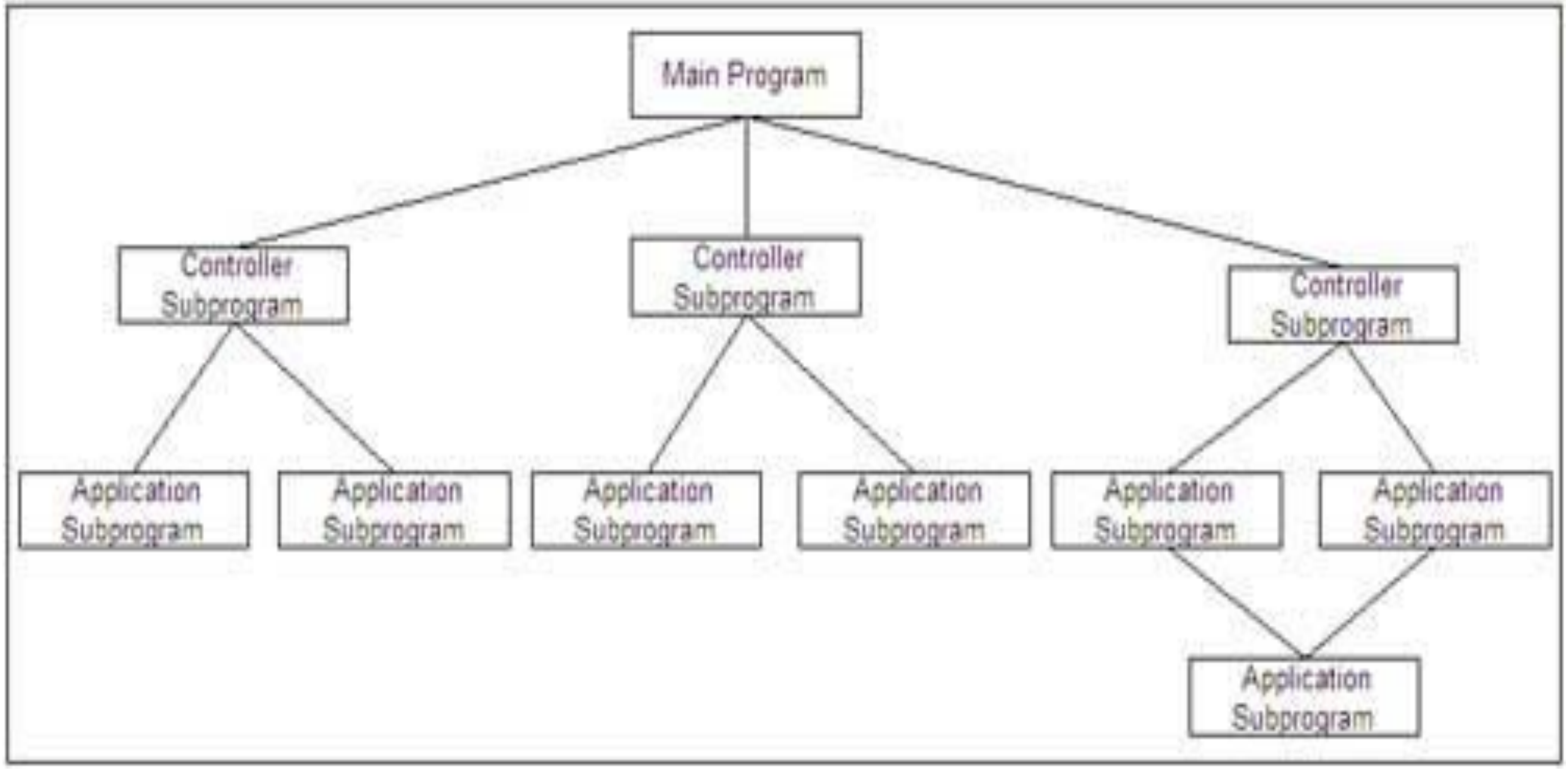
- A **software process model** is a simplified representation of a **software process**. Each **model** represents a **process** from a specific perspective. ... These generic **models** are abstractions of the **process** that can be used to explain different approaches to the **software** development.


Requirment analysis & specification
Design
Implimentation and unit testing
Integration and system testing
operaters and maintanance


Problem definition
Analysis
Design
Construction
Testing
Maintenance

Call-and return architecture

- Main program /sub program architecture:- Decompose function into a control hierarchy where a main program invokes a no. of program components, which in turn may invoke still other components.



Layered Architecture

- A no. of different layers are defined, each accomplishing that progressively become loser to m/c instruction set.
- At the outer layer, components service user interface operations
- At the inner layer, components perform operating system interfacing.

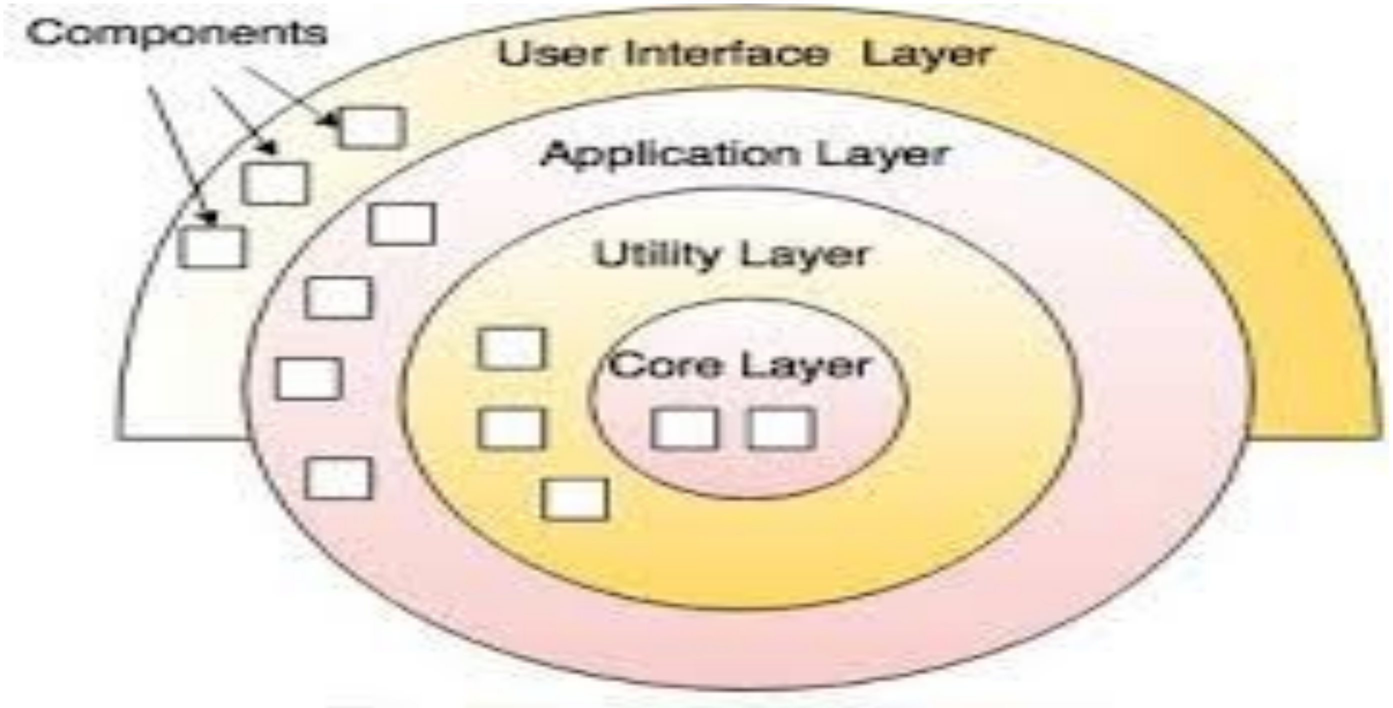
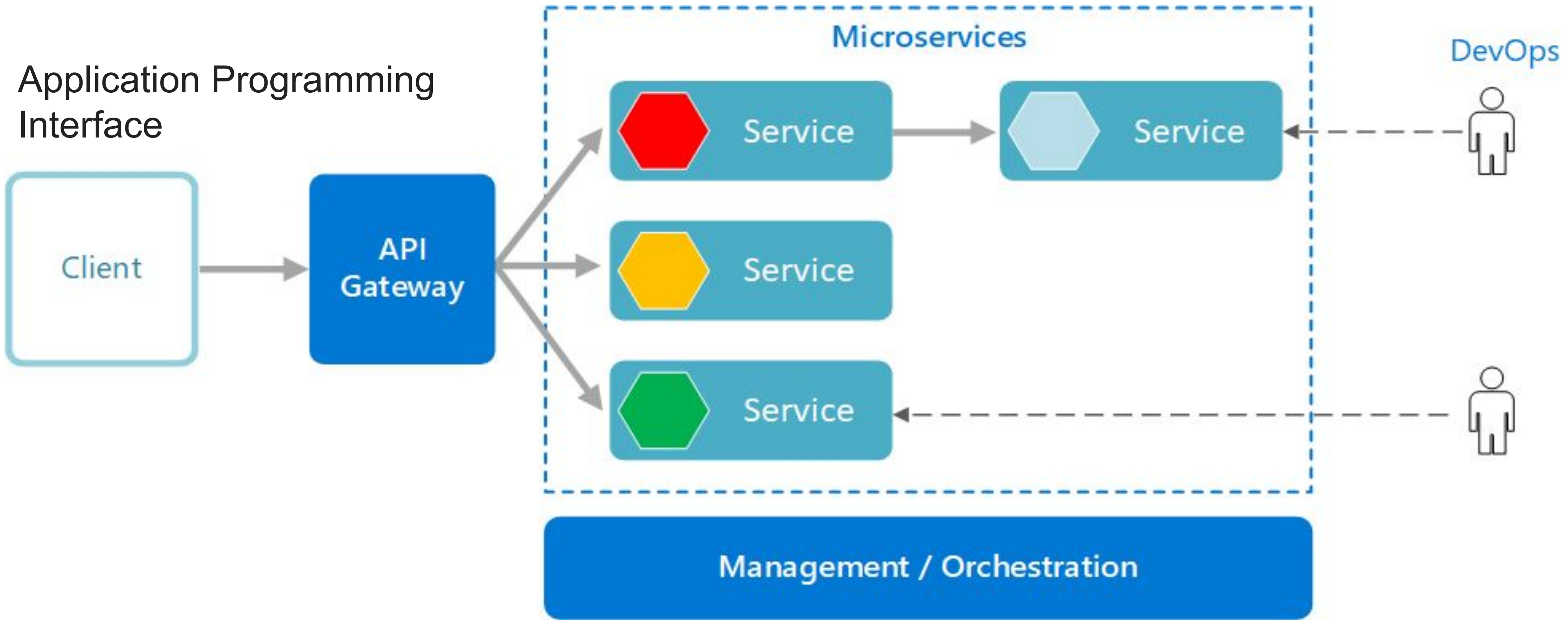


Fig.- Layered Architecture

Microservices architecture

- A microservices architecture consists of a collection of small, autonomous (independent) services. Each service is self-contained and should implement a single business capability



- What do you mean by DevOps?
- **DevOps** is the tools that increases an organization's ability to deliver applications and services at high velocity: evolving and improving products at a faster pace than organizations using traditional software development and infrastructure management processes.

- **What are microservices?**
- Microservices are small, independent, and loosely coupled. A single small team of developers can write and maintain a service.
- Each service is a separate codebase, which can be managed by a small development team.
-

  What defines a Microservice?
- **Microservices** - also known as the **microservice** architecture - is an architectural style that structures an application as a collection of services that are. Highly maintainable and testable. Loosely coupled. Independently deployable. Organized around business capabilities.

- **Benefits**
- **Agility** (quick)- Because microservices are deployed independently, it's easier to manage bug fixes and feature releases.
- **Small, focused teams-** A microservice should be small enough that a single feature team can build, test, and deploy it.

Reactive Architecture

- A **Reactive** system is an **architectural** style that allows multiple individual applications to coalesce (come together to form one mass or whole) as a single unit, reacting to its surroundings while aware of each other, and enable automatic scale up and down, load balancing, responsiveness under failure, and more.
- While the term reactive architecture has been around for a long time, only relatively recently has it been recognized by the industry and hit mainstream adoption. And the goal of this article is to analyze what reactive really is and why to adopt it.

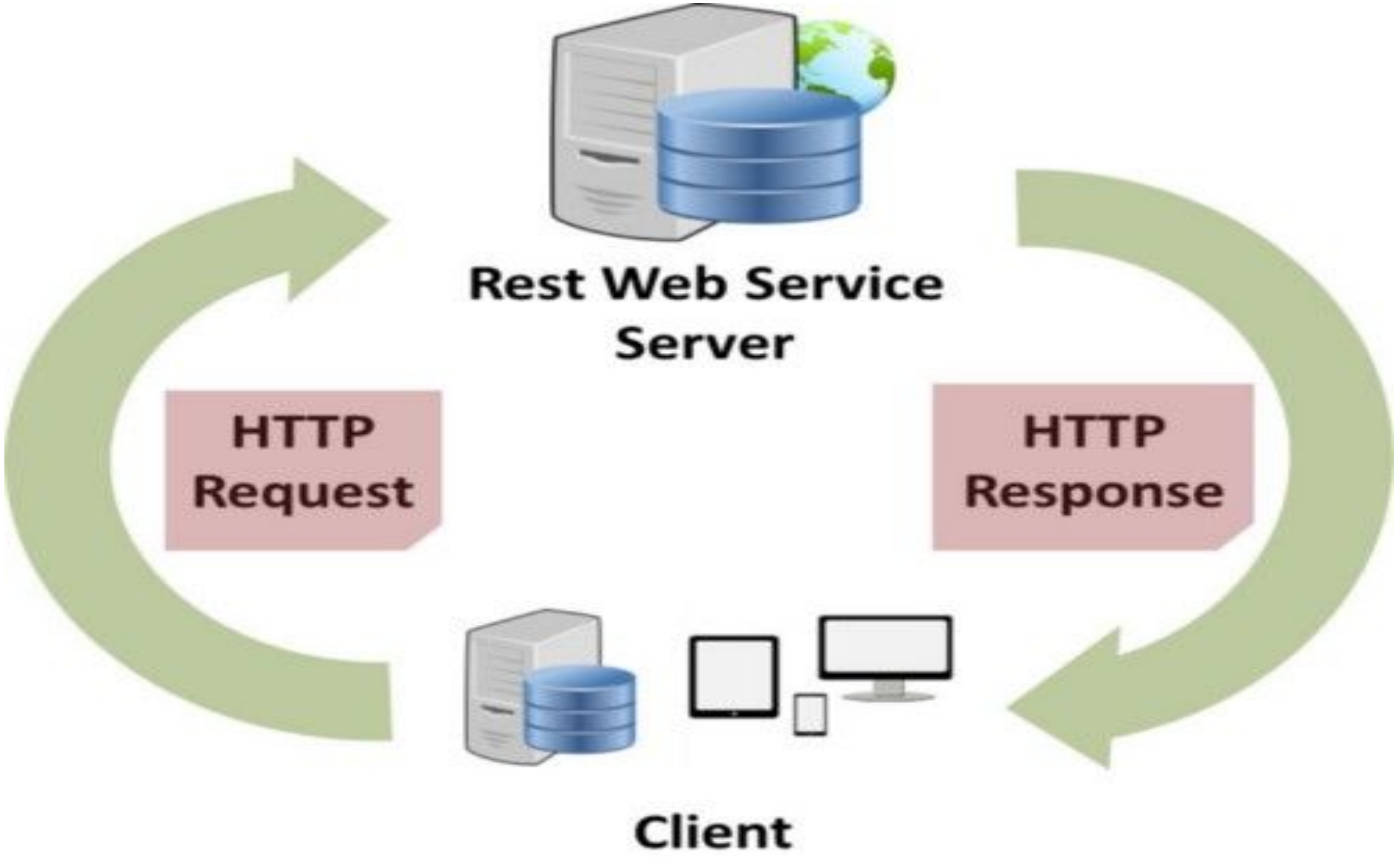
- Reactive Architecture is nothing more than the combination of reactive programming and software architectures. Also known as reactive systems, the goal is to make the system responsive, resilient, elastic, and message driven
- A Reactive system is an architectural style that allows multiple individual applications to coalesce as a single unit, reacting to its surroundings while aware of each other, and enable automatic scale up and down, load balancing, responsiveness under failure, and more.

- **Reactive Architecture Benefits**
- Be responsive to interactions with its users
- Handle failure and remain available during outages
- Strive under varying load conditions

Representational state transfer architecture

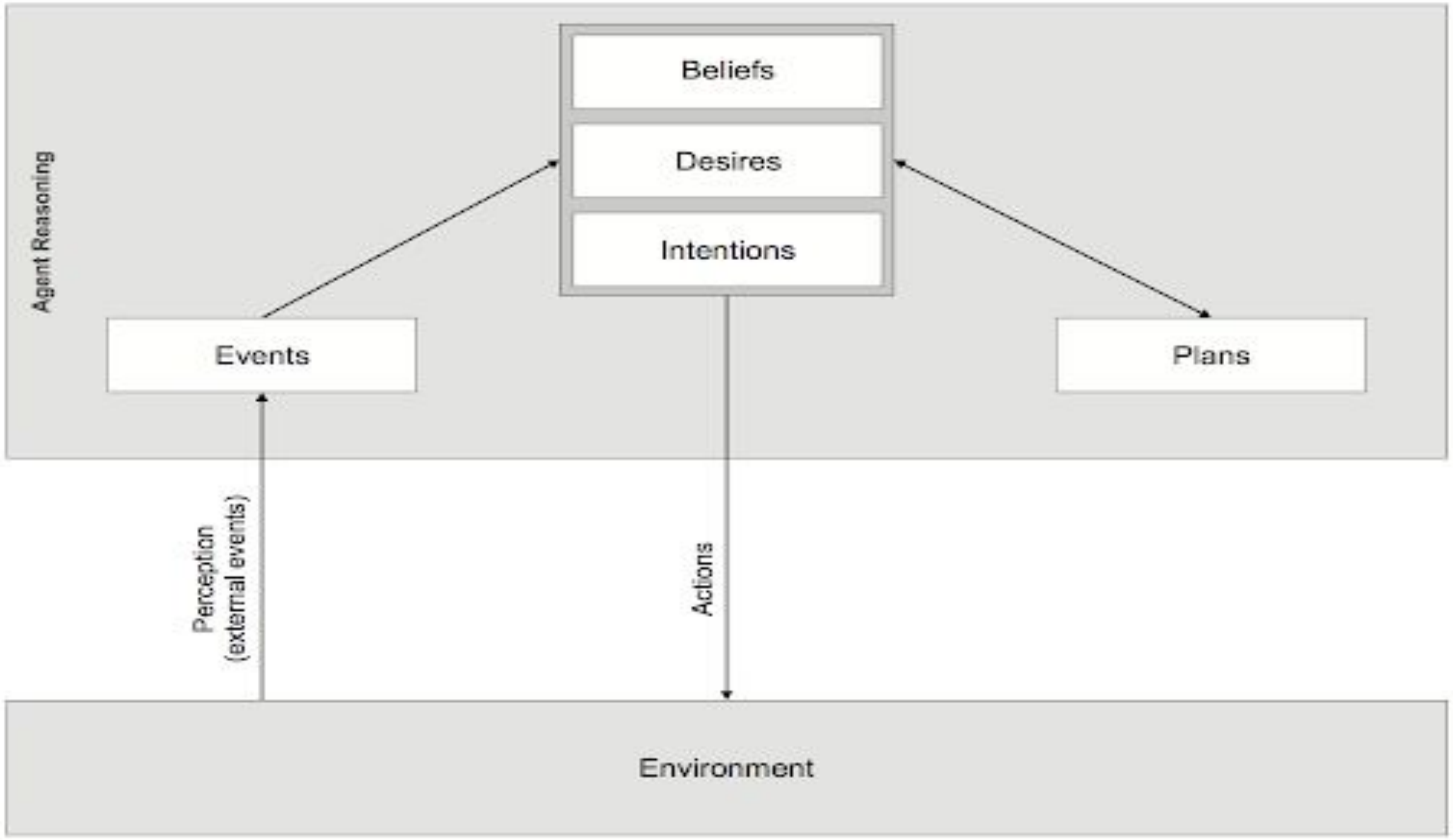
- **Representational state transfer** (REST) is a software **architectural** style that defines a set of constraints to be used for creating Web services. Web services that conform to the REST **architectural** style, called RESTful Web services, provide interoperability between computer systems on the internet.
- **What is REST ?**

  REST is an abbreviation for Representational State Transfer.

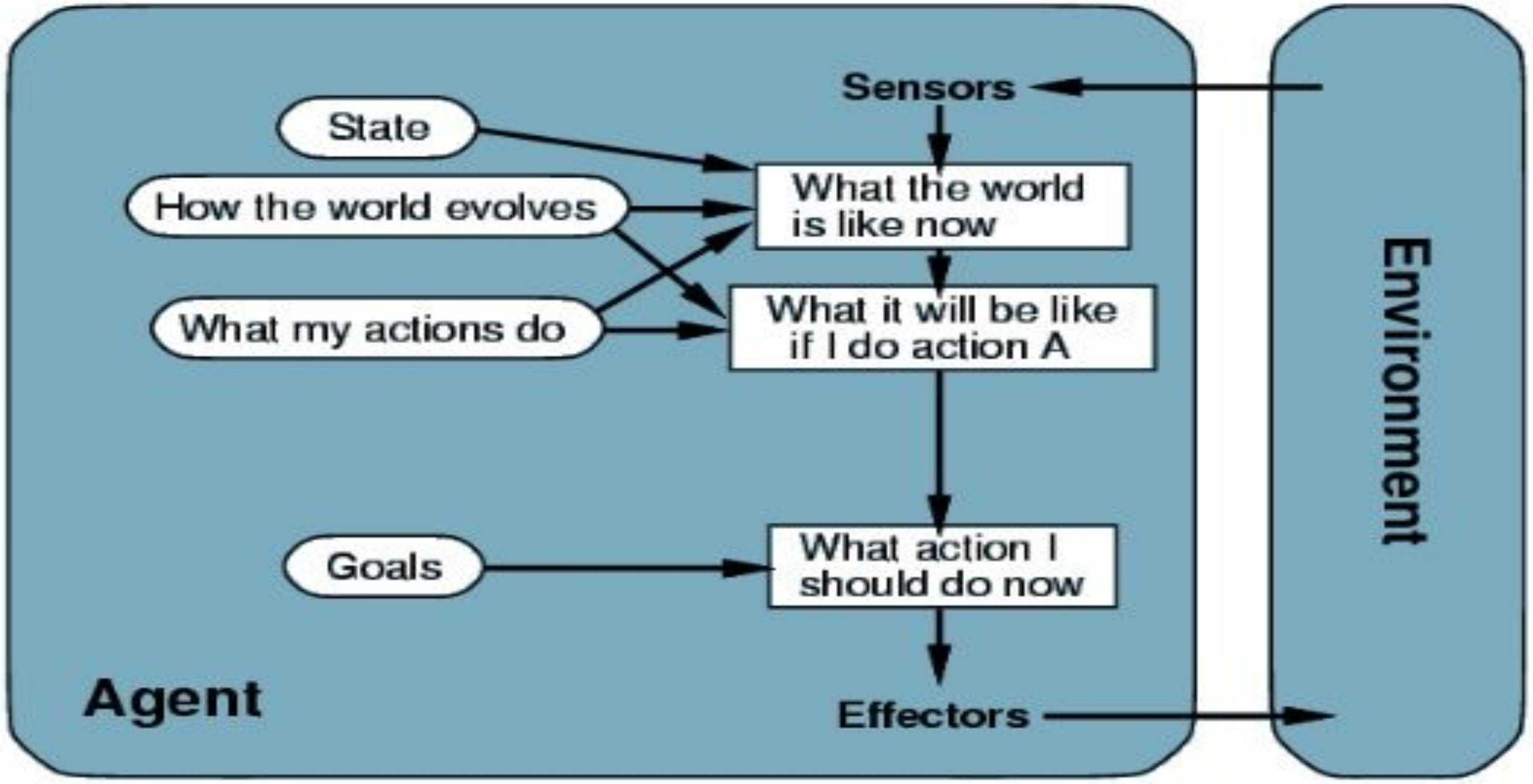

Rest Web Service Server
HTTP Request
HTTP Response
Client

- **What is the REST style?**
- REST is often described as an architecture style.
- It can be described as Set of formal and informal guides to creating architectures — “constraints”
- Client-server
- Stateless
- Cacheable
- Uniform interface
- Layered system
- Code on demand (optional)

- Agent architecture has been one of the core components in building an agent application. Agent architecture is considered as the functional brain of an agent in making decision and reasoning to solve problem and achieving goals.



Architecture of Goal Based Agent



Sensors
State
What the world is like now
How the world evolves
What it will be like if I do action A
What my actions do
Goals
What action I should do now
Agent
Effectors
Environment