SOFTWARE ARCHITECTURE

Unit 2

Architectural Styles

* **Architectural Style** - a named collection of architectural design decisions that are applicable in a given development context.

* **Architectural Pattern** - a named collection of architectural design decisions that are applicable to a recurring design problem, parameterized to account for different software development contexts in which that problem appears.

* Architectural Styles
  - Pipe and Filter
  - Data Abstraction & Object Orientation
  - Data Repositories
  - Blackboard Architecture
  - Table-driven Interpreters
  - Distributed / Heartbeat Architectures
  - Domain specific architectures
  - Process control systems
  - Heterogenous Architectures

(1) Pipe and Filter — def in unit 4

* **Pipelines** : restrict pipe and filter topologies to linear ones

Bounded Pipes : restrict the amount of data that can reside on a pipe

Typed Pipes - requires that the data passed between 2 filters have a well-defined type

* Advantages of Pipe and Filter

(i) supports reuse

(ii) can be easily maintained and enhanced

(iii) permits specialized analysis - throughput & deadlock analysis

* Data Flow in Pipe and Filters

(i) Push Only - write only

(ii) Pull Only - read only

(iii) Push / Pull

* Types of Filters

(i) Active Filters - An active filter pulls in data and pushes out the transformed data - it works with a passive pipe that provides read / write mechanisms for pulling and pushing. (eg. pipe & filter in Unix)

(ii) Passive Filter - lets connected pipes push data in and pull data out. - provides the read / write mechanisms.

(2) Data Abstraction and Object Orientation Types

→ Data representations and their associated primitive operations are encapsulated in an abstract data type.

→ Objects interact through function and procedure invocations

Two important aspects of this style are:

(i) the object is responsible for preserving the integrity of its representation

(ii) the representation is hidden from other objects

Advantages - change implementation without affecting clients

- allows designers to decompose problems into collections of interacting agents

Disadvantages - In order for one object to interact with another, it must know the identity of that other object

- whenever the identity of an object changes, it is necessary to modify all other objects that explicitly invoke it

(3) Data Repositories

→ A central data structure represents the current state, and a collection of independent components operate on the central data store.

eg. can use a blackboard data structure

(draw pic of class diagrams + fetch from DB)

→ suitable for large, complex information systems where many software component clients need to access them in different ways.

→ Requires data transactions to drive the flow of computation

Advantages : (i) data integrity
(ii) system scalability & reusability
(iii) reduces overhead

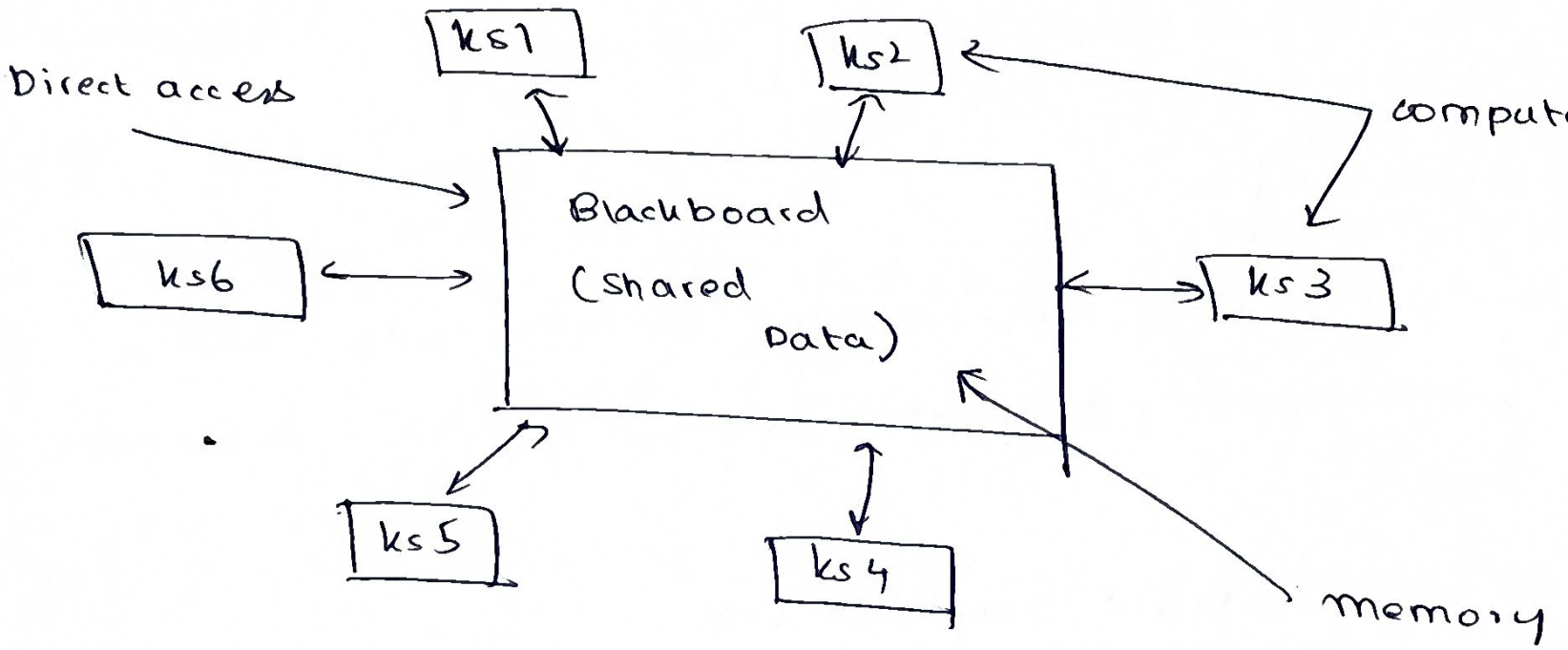
limitations : (i) vulnerability to failure
(ii) high dependency
(iii) data evolution is more difficult & expensive

(4) Blackboard Architecture

→ analogy of students and teachers working to solve a problem using a blackboard

→ has a blackboard, which is a central data structure where all data and intermediate results are stored and shared.

→ also has knowledge sources which are independent modules or components that contribute expertise to solve parts of the problem by reading from and writing to the blackboard.



(5) Table-Driven Interpreters

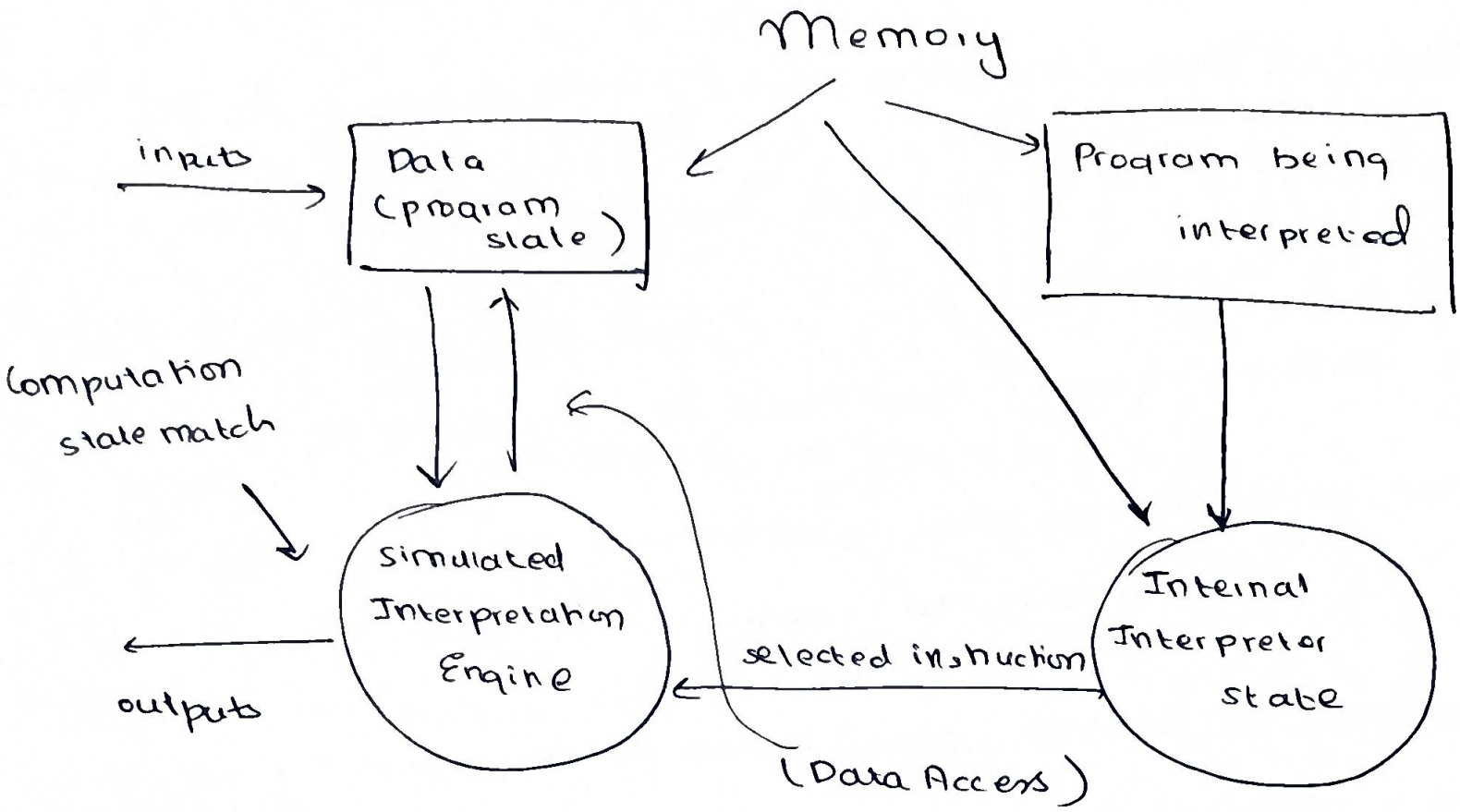
→ commonly used to build virtual machines, that close the gap between the computing engine, expected by the semantics of the program and the computing engine available in hardware

→ Thus, an interpreter has 4 components:

1. An interpretation engine to do the work
2. A memory that contains the pseudo-code to be interpreted.

3. a representation of the control state of the interpretation engine

4. a representation of the current state of the program being simulated



⑥ **Distributed Architectures** → Heartbeat Architectures

→ For any process in a network that initially only knows its neighbors, it is possible to get to know the entire network graph by means of network-wide message passing.

→ For this purpose, the concept of heartbeat algorithms is developed.

→ The heartbeat algorithm uses the FIFO nature of the signals sent across the network to ensure that all messages have been received, and that events can be properly ordered.

⑦ Domain-specific software architectures

⑧ Process Control Systems

→ Systems intended to provide dynamic control of a physical environment

→ These systems have feedback loops, - analogous to RL

helps determine a set of outputs that will produce a new state of the environment

A process control system must have the following:

(a) Controlled variable - a target - controlled variable - has a set point to reach

- The controlled variable data should be measured by sensors as feedback reference to recalculate manipulated variables

(b) Input variable

(c) Manipulated variable - can be adjusted by the controller

⑨ Heterogenous Architecture

combine architectures through → hierarchy

→ have a single component have a mixture of styles.

Look at UML diagram - Structural
- Behavioral
- Class diagram

- component diagram
- activity diagram
- sequence diagram
- state diagram
- interaction diagram