Presentation-Abstraction-Control – (Bengaluru Network Traffic Management)

# Definition:

The PAC architectural pattern defines a structure for interactive software systems in the form of a hierarchy of cooperating agents. Every agent is responsible for a specific aspect of the application's functionality and consists of three components: presentation, abstraction, and control.

## Structure:

---------------------------------------------------TOP LEVEL AGENT-------------------------------------------------------

ABSTRACTION

PRESENTATION

CONTROLLER

---------------------------------------------------INTERMEDIATE LEVEL AGENT-------------------------------------------------



---------------------------------------------------BOTTOM LEVEL AGENT-------------------------------------------------



### PAC mainly consists of 3 layers:

1. Top Level Agent: Maintains the global data that it can share with intermediate and bottom level Agents.
2. Intermediate Level Agent: Its main role is to do a composition of Bottom Level Agents and maintains synchronization between Top and Bottom Level Agent.
3. Bottom Level Agent: Do a specific semantic job. Focus on only one area.

### Components of Agents:

It has mainly 3 components:

1. Presentation: It’s similar to the View of MVC. Mainly consist of UI related code.
2. Abstraction: Business logic and the Data management such as inserting updating the record to the global repository can be handled at this layer.
3. Controller: Similar to the controller of MVC. It helps in creating link between different levels of Agents and also maintains a link inside a given Agent between presentation and abstract component.

OMT DIAGRAM – (BENGALURU NETWORK TRAFFIC MANAGEMENT)

BENGALURU TRAFFIC REPOSITORY

TRAFFIC MANAGEMENT REPOSITORY

**TOP LEVEL AGENT**

BENGALURU TRAFFIC VIEW COORDINATOR

**INTERMEDIATE LEVEL AGENT**

K R PURAM VIEW

M G ROAD VIEW

SENSOR 3

HOODI CIRCLE VIEW

SILK BOARD VIEW

SENSOR 2

SENSOR 1

**BOTTOM LEVEL AGENT**

SCENARIO 1 – UPDATE THE TRAFFIC DATA FROM SENOR 1, 2 AND 3 TO CORE REPOSITORY VIA TOP LEVEL AGENT

`

RECIEVMESSAGE GETDATA

CHANGE

CJA

RECIEVMESSAGE CHANGE

CHANGE

CJA

RECIEVMESSAGE CHANGE

CHANGE

CJA

RECIEVMESSAGE CHANGE

CHANGE

CJA

GETDATA

SENDMESSAGE

SETDATA

RECIEVEMESSAGE(SETDATA)

ENTER DATA

VIEW COORDINATOR

SILK BOARD VIEW

CONTROL

ABSTRACTION

TOP LEVEL AGENT

SENSOR-1 BENGALURU TRAFFIC REPOSITORY

1. Sensor 1 calls the controller of top level agent to insert the records in core repository.
2. Controller then invokes top level abstraction to update the records in database.
3. Abstraction then calls the controller again so that it can then send messages to bottom level controller via intermediate
4. Bottom level Abstraction then does its specific task and display the view with latest data.

SCENARIO 2: USER CALLS SILK BOARD VIEW TO SEE THE TRAFFIC UPDATE.

1. Users ask for a Silk Board view.
2. Intermediate level controller gets the appropriate Presentation from Bottom level agent which is Silk Board.
3. Silk Board view (Abstraction) gets latest data from Top Level Agent via Intermediate level controller.
4. Once the latest data is received the controller of Silk Board View gets this data from its Abstraction and passes it on to its Presentation.
5. The Silk Board Presentation then display the latest data to the user via is UI.

# Benefits:

Separation of concern: Every agent does its own work. They are loosely coupled and highly cohesive.

Modifiability: Any new agent can be added without disturbing the existing Agents at any level.

Multitasking: Multiple process or threads can run different agents thereby increasing Performance of the system.

Usability: Increase the usability of the system as the Top level contains the basic UI while the lower levels provide enhancement to specific parts of the UI such as providing facility to zoom in and out, etc.

Liability:

System becomes complex over the time with multiple agents at different levels.