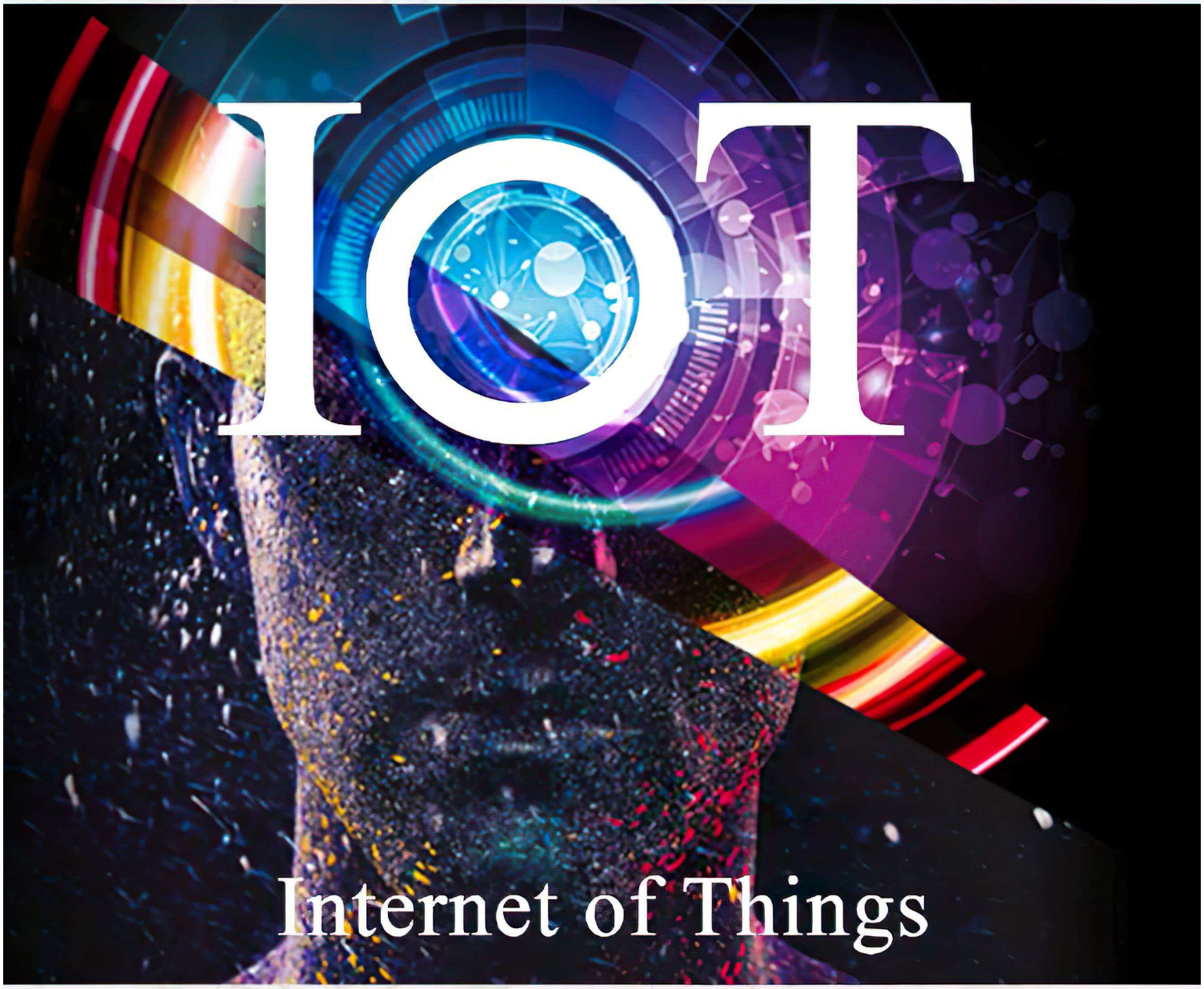
Jump2Learn
PUBLICATION

www.jump2learn.com



Internet of Things

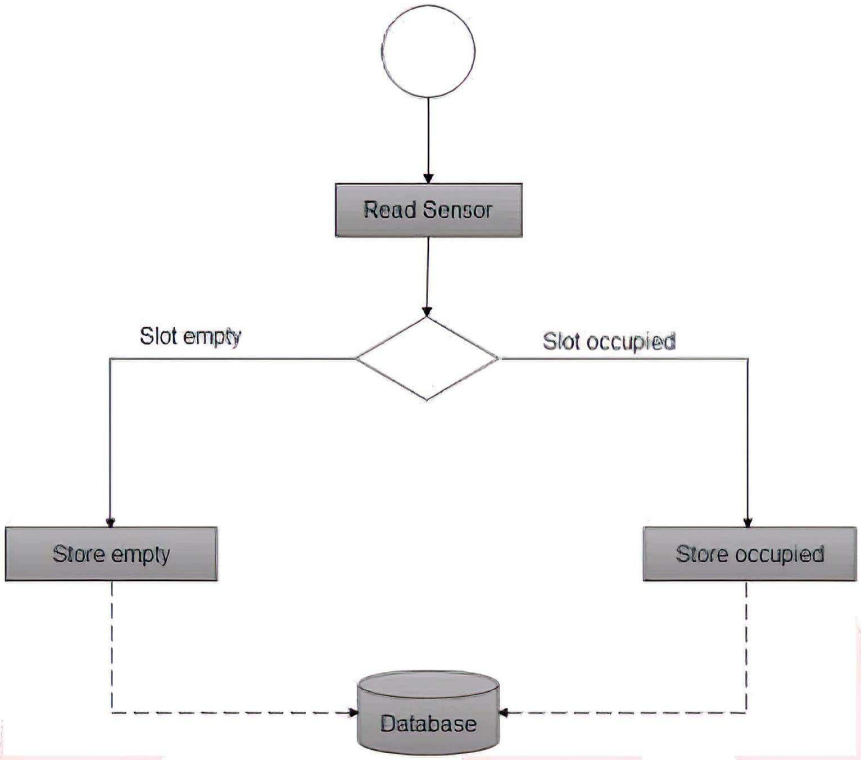
Jump2Learn - The Online Learning Place

Mr. Vijesh Shukla | Ms. Khushbu A. Patel | Mr. Ankit R. Patel

Unit - 5

Case Study

5.1 IoT for Smart City Application



The number of urban residents is growing by nearly 60 million every year. In addition, more than 60 percent of the world's population will be living in cities by 2050.As a result, people occupying just 2 percent of the world's land will consume about three-quarters of its resources. Moreover, more than 100 cities of 1 million people will be built in the next 10 years.

There are different types of application of smart cities like Smart environment, Smart parking, Smart living, Smart Governance, Smart Waste Management and Smart Street lighting etc.

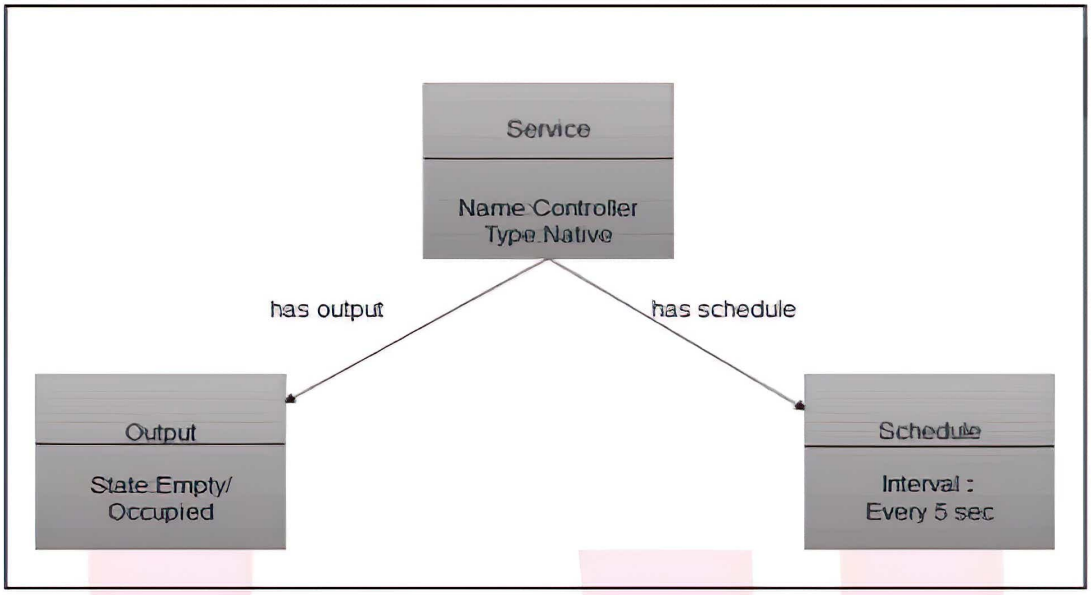
Smart Parking

The purpose of a smart parking system is to detect the number of empty parking slots and send the information over the internet to smart parking application backend. These applications can be accessed by drivers from smartphones, tablets or from in-car navigation systems.in smart parking, sensors are used for each parking slot, to detect whether the slot is empty or occupied. This information is aggregated by a local controller and then sent over the internet to a server.

The following figure shows the domain model for the smart parking system. The domain model includes a physical entity for the parking slot and the corresponding virtual entity.

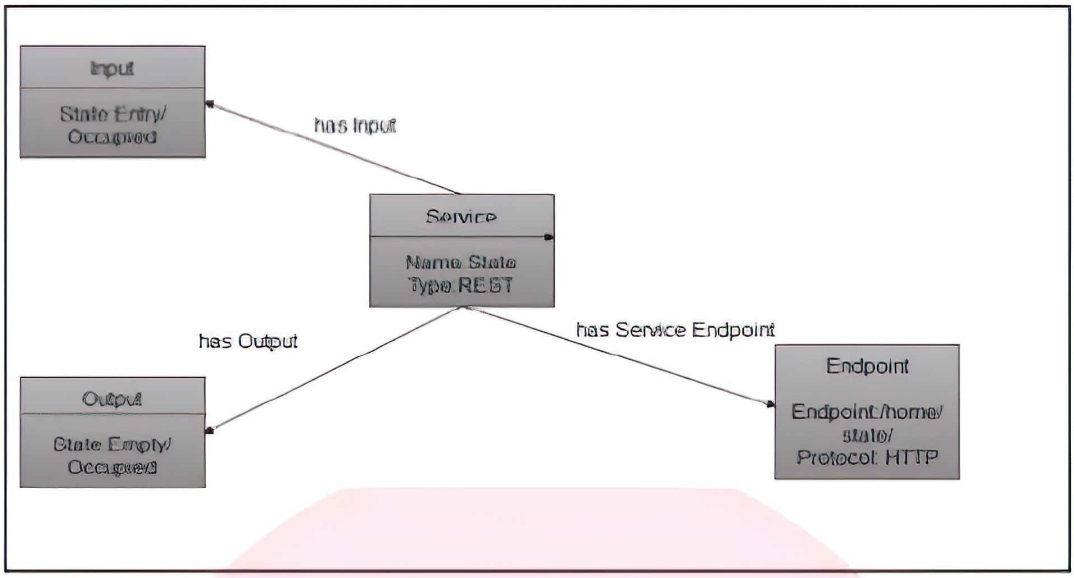
Process Specification for the smart parking IoT System

Above figure shows the process diagram for the smart parking system. Each parking slot has an ultrasonic sensor fixed above which can detect the presence of a vehicle in the slot. Each sensor is read at regular intervals and the state of the parking slot (empty / occupied) is updated in database.



Controller service of the smart parking IoT system

Above figure show specification for the controller services of the smart parking system here in Schedule part, every 5 seconds sensors are used for each parking slot, to detect whether the slot is empty or occupied. In output part, State will be changed based on the detection of the presence of a vehicle in the slot.

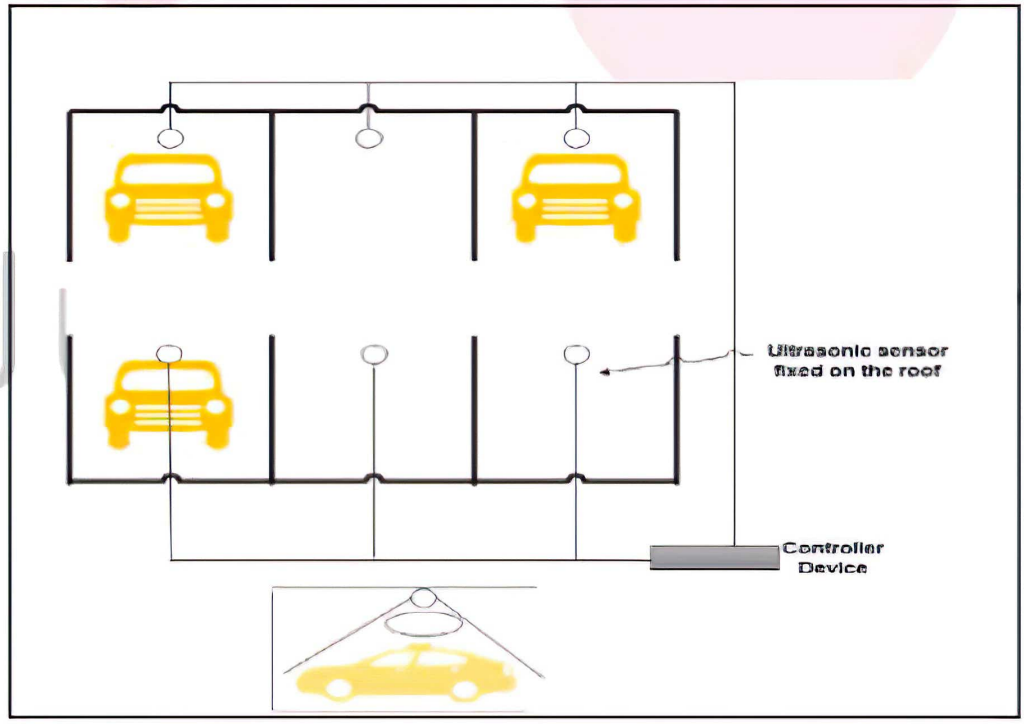


Service specification for the smart parking IoT system - State service

In above figure show specification of the state services of the smart parking system. The services are derived from the process specification and the information model. The smart parking system has two services.

(1) A service that monitors the parking slots (using ultrasonic sensors) and updates the status in the database on the cloud.

(2) A service that retrieves the current state of the parking slots.



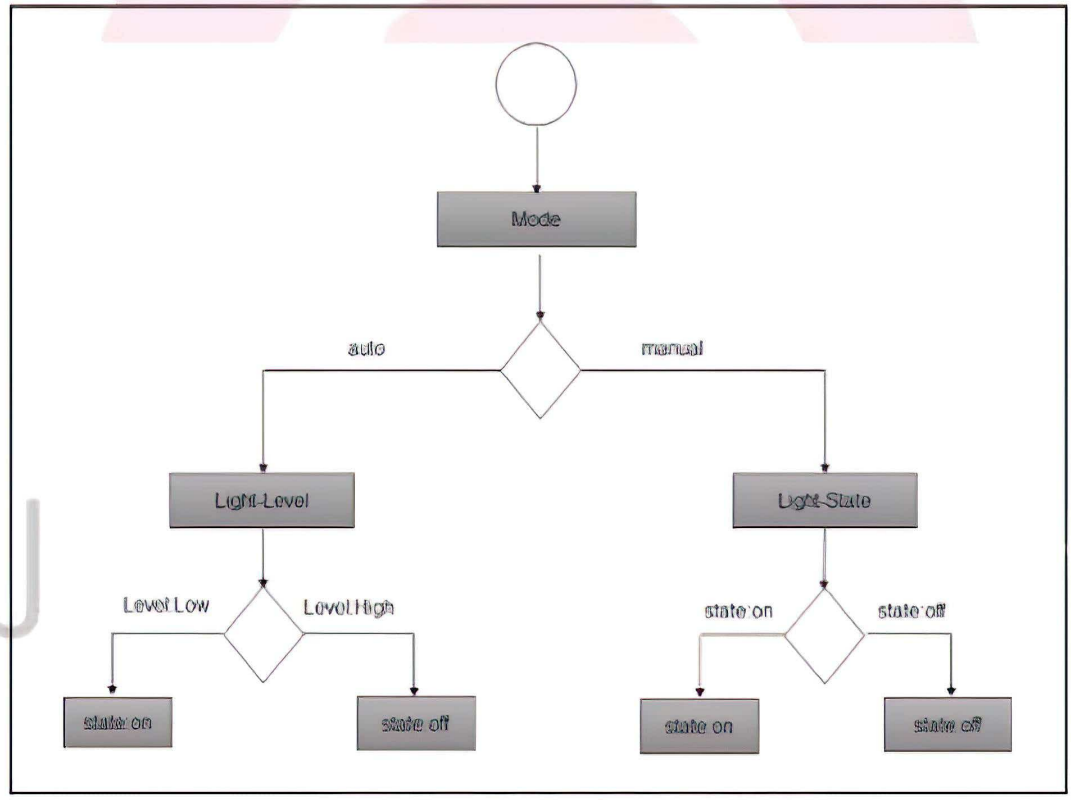
Deployment of sensors for smart parking system

Above figure shows how the sensor deployed in a roof of parking area. The View Sets for the model (StateViewSet) are included in the views file. The home view renders the content for the smart parking application home page that displays the status of the parking slots. Notice that a request is sent to the REST(Representational State Transfer) service to obtain the state of a parking slots. Notice that a request is sent to the state REST service to obtain the state of a parking slot.

5.2 IoT for Smart Home

Smart Lighting

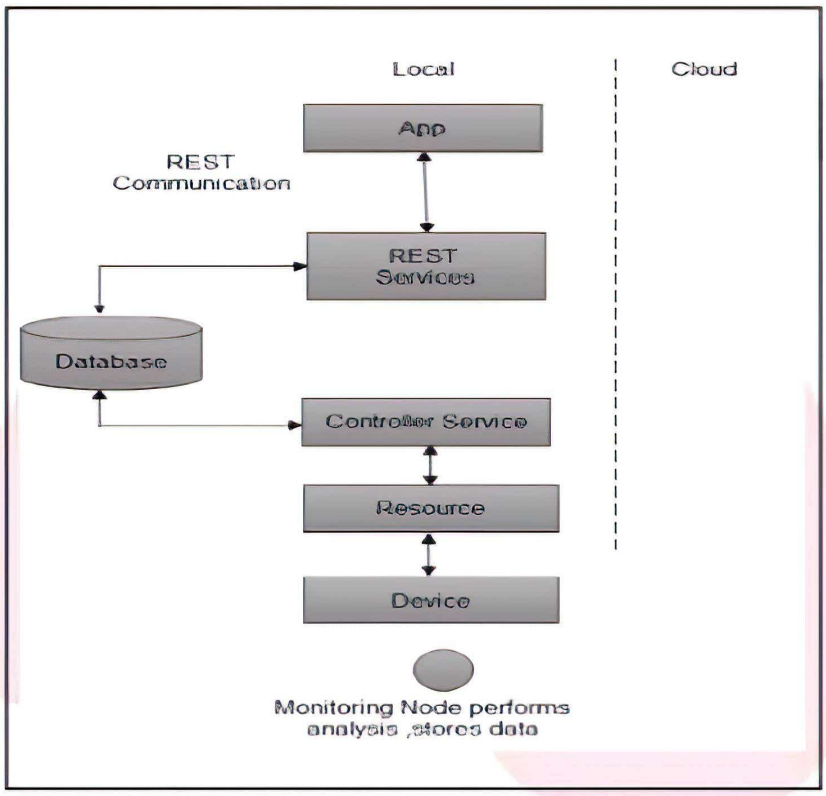
The purpose of the home automation system is to control the lights in atypical home remotely using a web application. The system includes auto and manual modes. In auto mode, the system measures the light in a room and switches on the light when it gets dark. In manual mode , The system provides the option of manually and remotely switching on/off the light.



Process specification for Smart Lighting

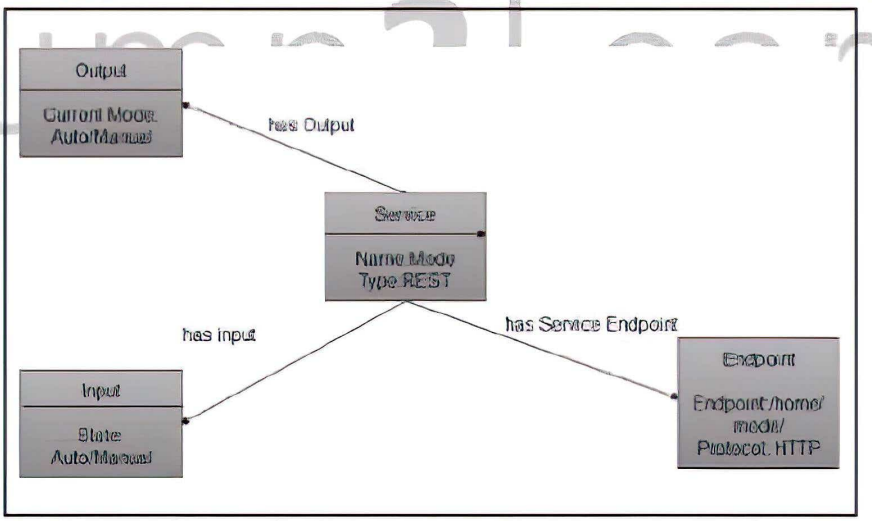
Above figure shows the process diagram for the smart lighting system. The process diagram shows the two modes of the system-auto and manual. In the process diagram the circle denotes the start of a process, diamond denotes a decision box and rectangle denotes a state

or attribute. When the auto mode is chosen, the system monitors Light level. If the light level is low, the system changes the state of the light to "off". When the manual mode is chosen, the system checks the light state set by the user. If the light state set by the user is "on". The system changes the state of light to "on". Whereas, if the light state set by the user is "off", the system changes the state light to "off".



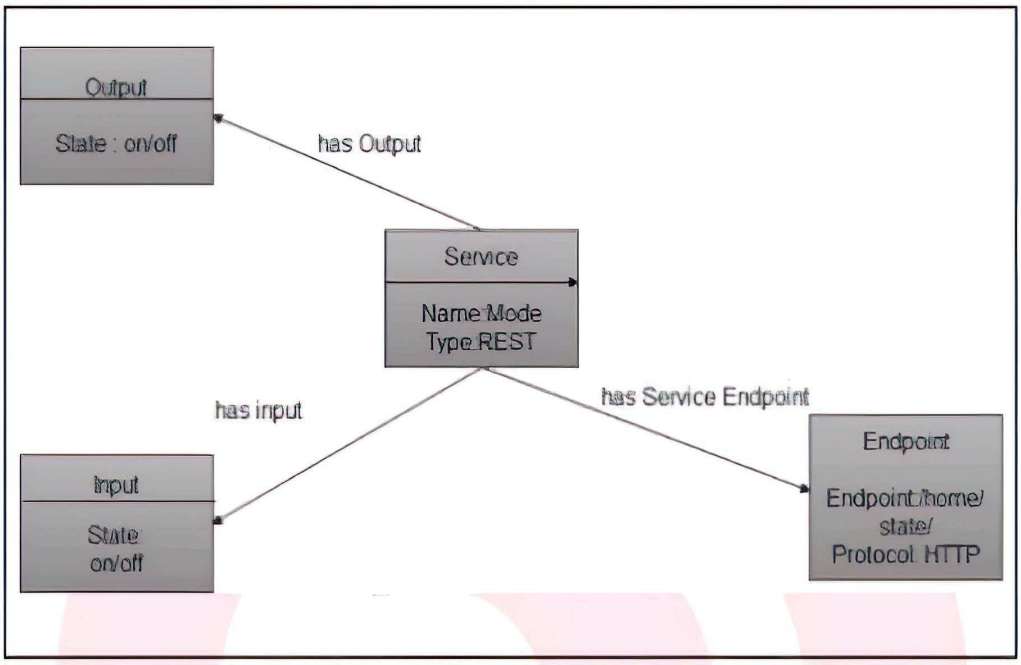
Deployment design of the home automation IoT system.

Above figure shows the deployment design of the home automation system. The system has two REST services (mode and state) and a controller native service. The following two figure show specifications of the mode and state REST services of the home autonomous system.



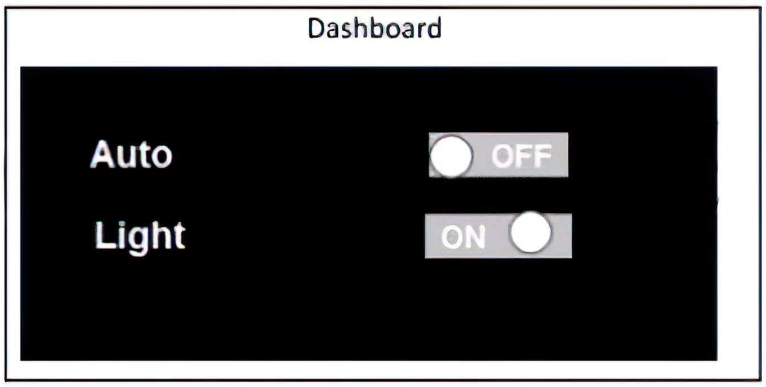
Service specification for home automation IoT system-mode service

The mode service is RESTful web service that sets mode to auto or manual, or retrieves the current mode. The mode is updated to/ retrieved from the database.



Service specification for home automation IoT system – State service

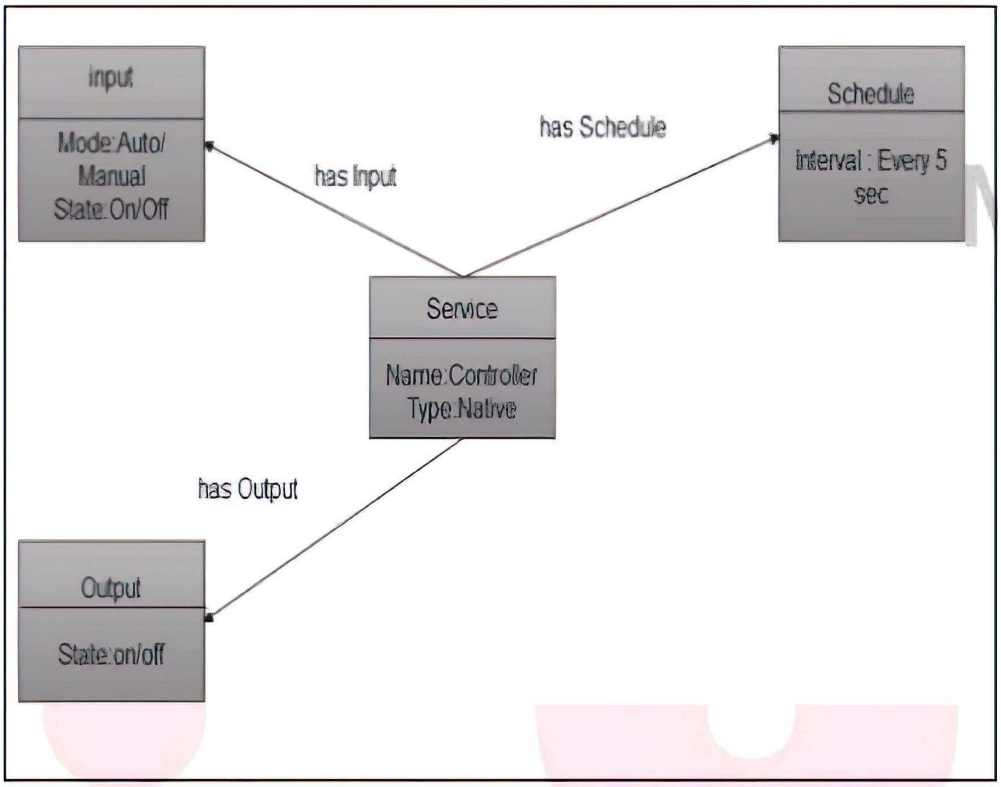
The state service is a RESTful web service that sets the light appliance state to on/off, or retrieves the current light state. The state is updated to/retrieves the current light state. The state is updated to/retrieved from the status database.



Home automation web application screenshot

Above figure shows a screenshot of the home automation web application. Figure show two Switches one for Auto or Manual (ON/OFF) and second is for Light (ON/OFF).If we off Auto option then it will be convert into manual mode otherwise it is in Auto mode. The devices

and components are used in this example are Raspberry Pi minicomputer, LDR sensor and relay switch actuator.



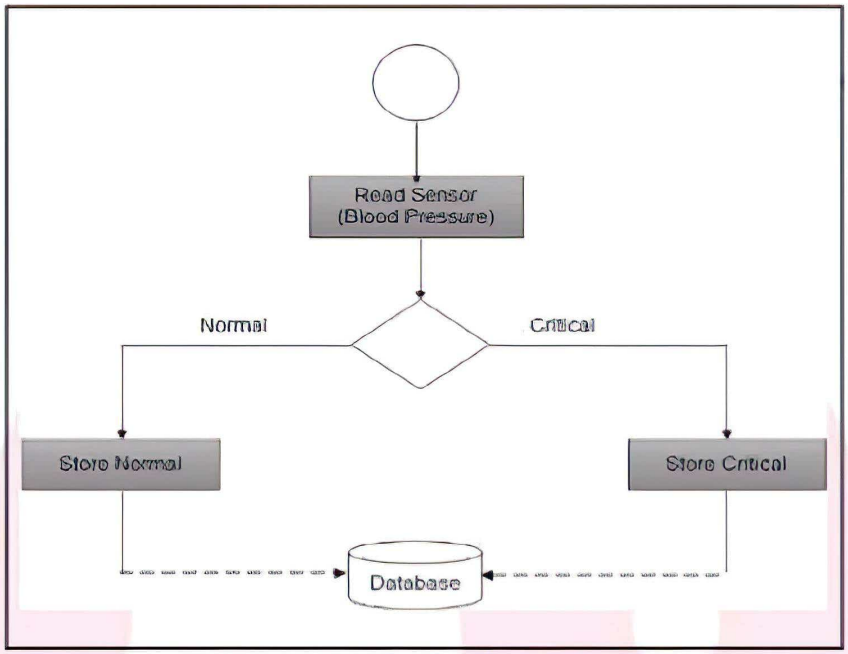
Controller services of the home automation IoT system

The above figure shows the specification of the controller native service that runs on Raspberry Pi. When in auto mode, the controller service monitors the light level and switches the light on/off and updates the status in the status database. When in manual mode, the controller service, retrieves the current state from the database and switches the light on/off.

5.3 IoT for Health & Lifestyle

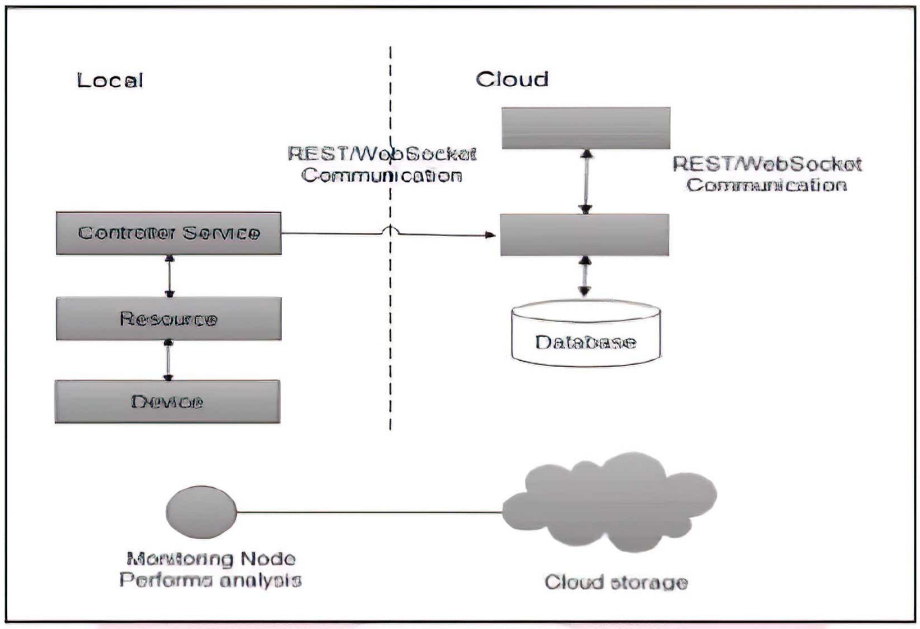
Basic purpose of Smart health & Lifestyle is some wearable IoT device that continuous monitoring of physiological parameters that can help in continuous health and fitness monitoring. These wearable devices may be in various forms such as belts and wrist-bands. The wearable devices form a type of wireless sensor networks called body area networks in which the measurements from a number of wearable device are continuous sent to a master node(such as smart-phone) which then sends the data to a server or a cloud based back –end

for analysis and archiving. Commonly used body sensor include: body temperature, heart rate, pulse oximeter oxygen saturation(SPo2), blood pressure, electrocardiogram(ECG) etc. Some challenges in the health care system are smarter hospital, Data Integration/realtimeness, Medical resource shortness, Bad Health habits, Lack of information sharing etc.



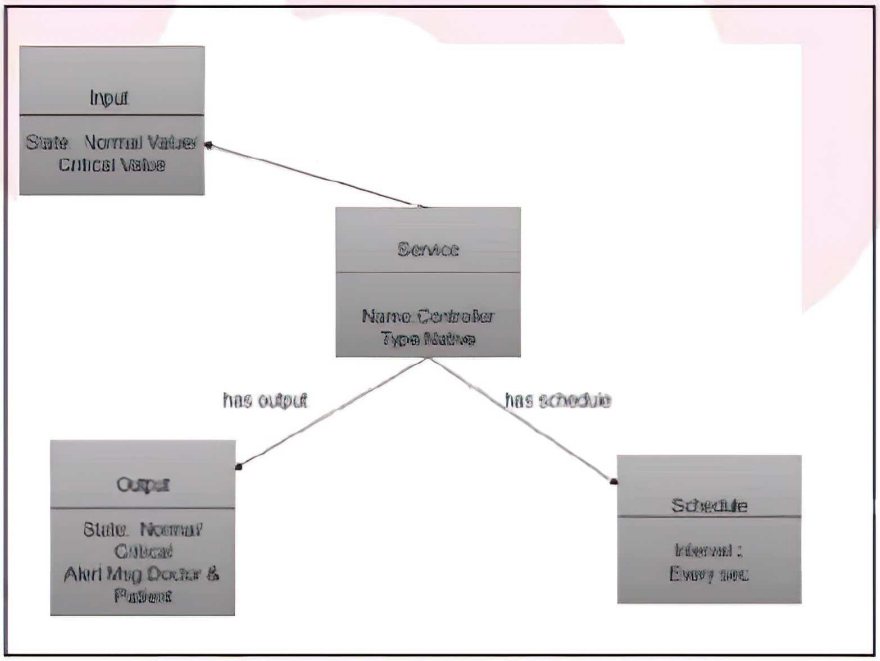
Process specification for IoT Health System

Above figure shows the process diagram for the IoT health system. The process diagram shows the two part Normal and Critical. In the Normal part, the blood pressure value is less than threshold_min value and in the Critical part, the blood pressure value is more than threshold_max value. In the process diagram the circle denotes the start of a process, diamond denotes a decision box and rectangle denotes a state or attribute.

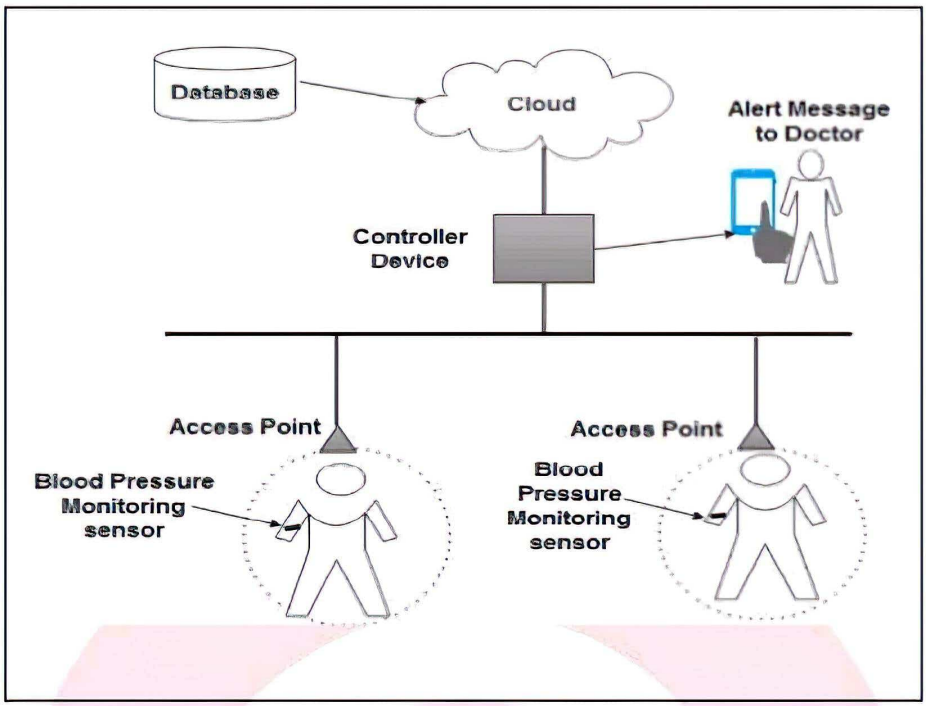


Deployment design for IoT Health System

Above figure shows the deployment design of the IoT Health system. The system has REST services and a controller native service.



Controller services of the IoT Health system



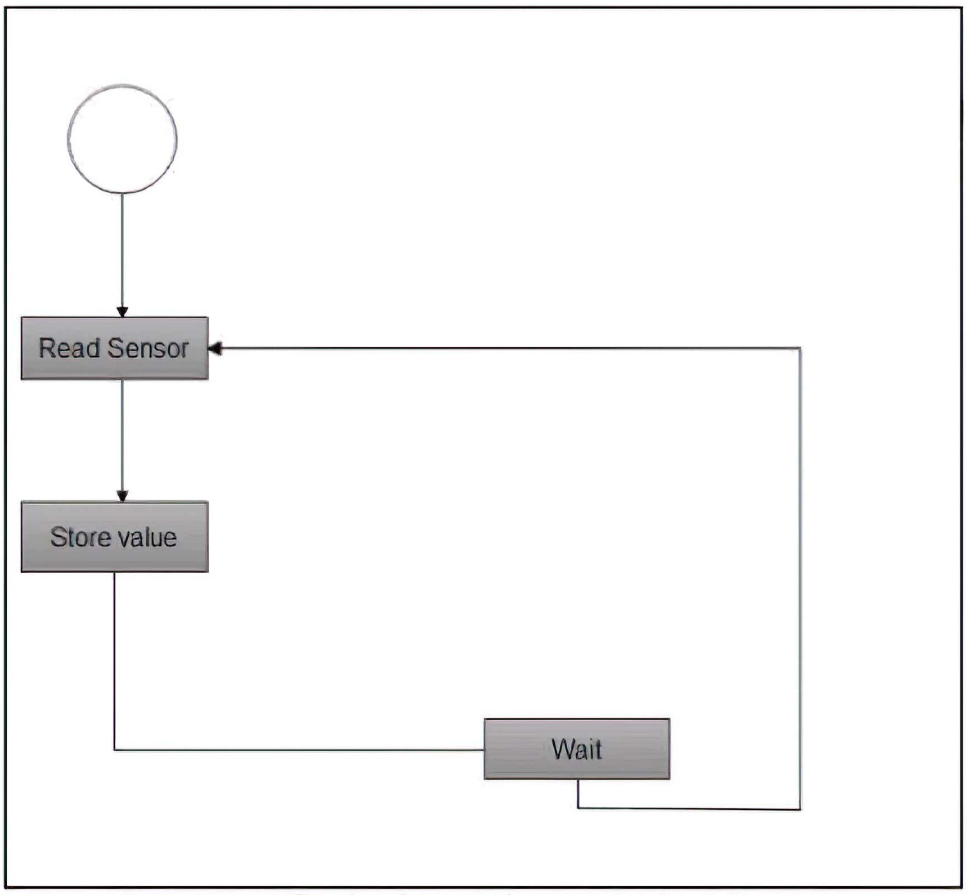
Deployment of Adhoc Wireless Sensor for Health Monitoring System

In above figure shows blood pressure monitoring adhoc wireless sensor attach to patient body that collect continues input value of blood pressure form the patient body and the controller service calls the REST service to store these measurements in the cloud. In the cloud application can analyze and check threshold minimum and threshold maximum value of blood pressure set previously according to doctor advice and display on dashboard also. If blood pressure value more than upper limit of threshold_max value or less than threshold_min value then alert message sent to the corresponding doctor and patient. Doctor can give advice to the patient immediately.

Case Study for Practice

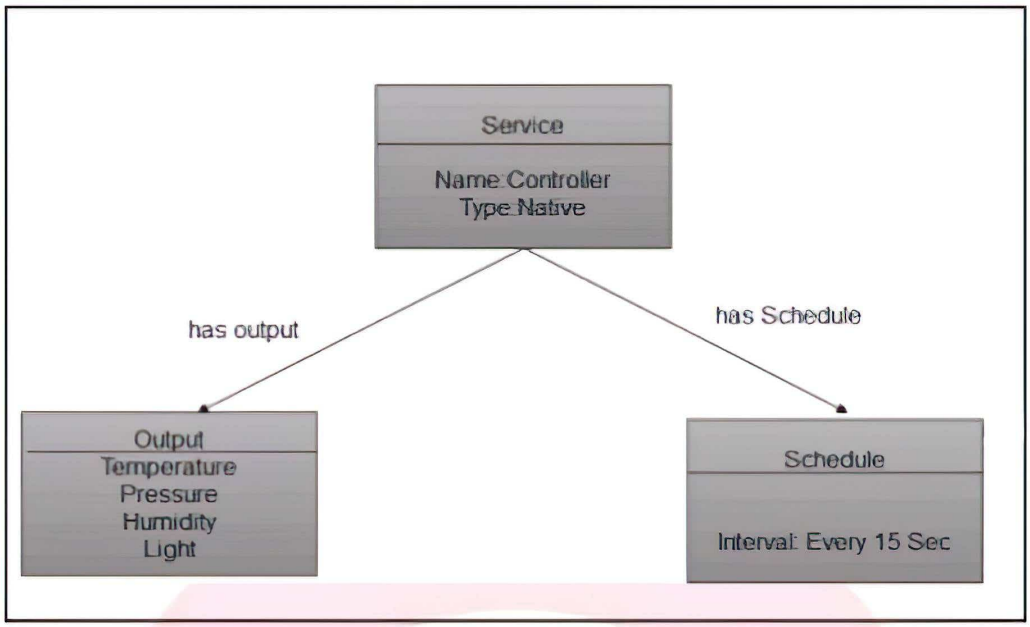
1. Weather Monitoring System

The purpose of the weather monitoring system is to collect data on environmental conditions such as temperature, Pressure, humidity and light in an area using multiple end nodes. The end nodes send the data to the cloud where the data is aggregated and analyzed.



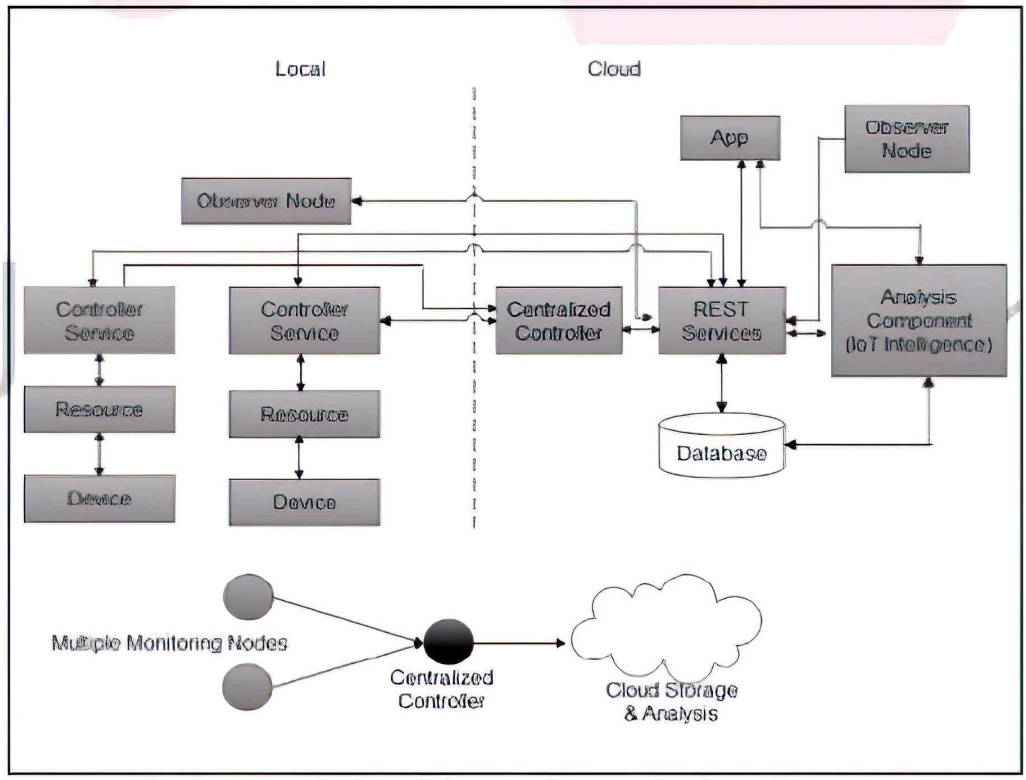
Process specification for weather monitoring IoT system

Above figure shows process specification for the weather monitoring system. The process specification shows that the sensors are read after fixed intervals and the sensor measurements are stored.



Controller service of the weather monitoring IoT system

Above figure shows the specification of the controller service for the weather monitoring system, The controller service runs as a native service on the device and monitors temperature, pressure, humidity and light once every 15 seconds. The controller service calls the REST service to store these measurements in the cloud.



Deployment design of the weather monitoring IoT system

The above figure shows the deployment design for the weather monitoring system. The system consists of multiple nodes placed in different locations for monitoring temperature, humidity and pressure in an area. The end nodes are equipped with various sensors. The end nodes send the data to the cloud and the data is stored in a cloud database. The analysis of data is done in the cloud to aggregate the data and make predictions. A Cloud – based application is used for visualizing the data. The centralized controller can send control commands to the end nodes, to configure the monitoring interval on the end node.

****