SNS COLLEGE OF TECHNOLOGY

Coimbatore-35
An Autonomous Institution

Accredited by NBA – AICTE and Accredited by NAAC – UGC with 'A++' Grade
Approved by AICTE, New Delhi & Affiliated to Anna University, Chennai

DEPARTMENT OF INFORMATION TECHNOLOGY

19ITT302 INTERNET OF THINGS

III YEAR – V SEM

UNIT 05

DESIGN METHODOLOGY & FUTURE TRENDS

Syllabus

UNIT I **IoT INTRODUCTION AND APPLICATIONS** **8**

Overview and Motivations - IPv6 Role - IoT Definitions - Observations - ITU-T Views – Working Definition - IoT Frameworks - Basic Nodal Capabilities – Physical Design of IoT - Logical Design of IoT – Applications:- City Automation Automotive Applications - Home Automation - IoT Levels & Deployment Templates - IoT and M2M

UNIT II **FUNDAMENTAL MECHANISMS & KEY TECHNOLOGIES** **8**

Identification of IoT Objects and Services- Structural aspects of IoT-Environment Characteristics Traffic Characteristics-Scalability-Interoperability-Security and privacy -Key IoT Technologies :Device Intelligence - Communication Capabilities - Mobility Support - Device Power –Sensor Technology -RFID Technology - Satellite Technology - IoT Enabling Technologies- WSN, Cloud computing, Big data Analytics, communication protocols, embedded systems

UNIT III **EVOLVING IoT STANDARDS & PROTOCOLS** **11**

IETF IPv6 Routing Protocol for RPL Roll – Constrained Application Protocol (CoAP) – Representational State Transfer (REST) – Third Generation Partnership Project Service Requirements for Machine Type Communications- Over Low Power WPAN (6LoWPAN)- IP in Small Objects (IPSO) - WPAN Technologies for IoT/M2M – ZigBee/IEEE 802.15.4, RF4CE,Bluetooth and its Low Energy Profile.

Syllabus

UNIT IV **IPv6 TECHNOLOGIES FOR THE IOT** **9**

Motivations - Address Capabilities - IPv6 Protocol Overview - IPv6 Tunneling - IPsec in IPv6 - Header Compression Schemes - Quality of Service in IPv6 - MOBILE IPv6 -Protocol Details - Generic Mechanisms - New IPv6 Protocol - Message Types - Destination Option - Modifications to IPv6 Neighbor Discovery - Requirements for Various IPv6 Nodes - Correspondent Node Operation - HA Node Operation-Mobile Node Operation Relationship to IPV4 Mobile IPV4(MIP)-IPV6 Over Low-Power WPAN-Goals-Transmission of IPV6 Packets Over IEEE 802.15.4.

UNIT V **DESIGN METHODOLOGY & FUTURE TRENDS** **9**

IoT System Management with NETCONF-YANG: Need for IoT Systems Management – Simple Network Management Protocol (SNMP) –Limitations of SNMP, Network Operator Requirements NETCONF-YANG-IoT Systems Management with NETCONF-YANG -IoT Platforms Design Methodology – IoT Physical Devices & Endpoints - Raspberry Pi- Linux on Raspberry Pi –Raspberry Pi Interfaces - Programming Raspberry Pi with Python - Designing a RESTful WebAPI – Amazon Web Services for IoT

TEXT BOOKS

1 Daniel Minoli, Building the Internet of Things with IPv6 and MIPv6: The Evolving World of M2M Communications, Wiley Publications, First Edition, 2013.

2 Arsheep Bahga, Vijay Madisetti, Internet of Things: A Hands-On Approach, Universities Press, First Edition, 2014.

REFERENCES

1 Jean-Philippe Vasseur, Adam Dunkels, Interconnecting Smart Objects with IP: The Next Internet, Elsevier Publications, 2010

2 Adrian McEwen, Hakim Cassimally, Designing the Internet of Things, Wiley Publications, First Edition, 2013.

3 N. Ida, Sensors, Actuators and Their Interfaces, SciTech Publishers, 2014.

DESIGN METHODOLOGY & FUTURE TRENDS

IoT System Management with NETCONF-YANG: Need for IoT Systems Management – Simple Network Management Protocol (SNMP) –Limitations of SNMP, Network Operator Requirements NETCONF-YANG-IoT Systems Management with NETCONF-YANG -IoT Platforms Design Methodology – IoT Physical Devices & Endpoints - Raspberry Pi- Linux on Raspberry Pi – Raspberry Pi Interfaces - Programming Raspberry Pi with Python - Designing a RESTful WebAPI – Amazon Web Services for IoT

IoT System Management with NETCONF-YANG

Need for IOT Systems Management

IoT systems have complex software, hardware (sensors, actuators), network resources, data collection, analysis services, communication protocols, and user interfaces.

The need for managing IoT systems are:

1. Automating Configuration:

· System management interfaces provide predicate and easy-to-use management capability to automation system configuration when a system consists of multiple devices or nodes.
· Ensures all devices have the same configuration and variations or errors due to manual configurations are avoided.

2. Monitoring Operational & Statistical Data:

· Operational data:- the system's operating parameters that are collected by the system at runtime.
· Statistical data:- system performance (e.g. CPU and memory usage) data for fault diagnosis or prognosis (forecasting).

IoT System Management with NETCONF-YANG

3. Improved Reliability:

· By validating the system configurations before use.

4. System-Wide Configuration:

· IoT systems consist of multiple devices or nodes, which have wide system configurations for the correct functioning.

· Each device is configured separately (either manual or automated).

· Used in system faults or undesirable outcomes.

· Ensures that the configuration changes are either applied to all devices or to none.

· In the failure, the configuration changes are rolled back.

5. Multiple System Configurations:

· Some systems have multiple valid configurations according to different times or in certain conditions.

IoT System Management with NETCONF-YANG

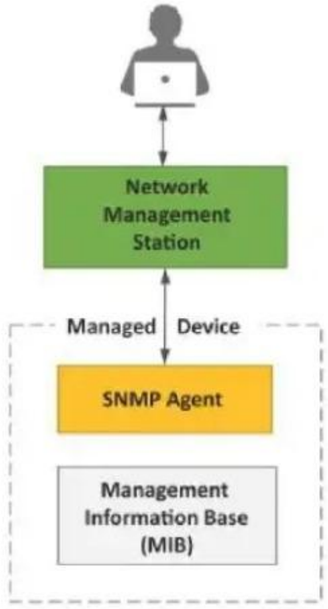
Retrieving & Reusing Configurations:

- Help in reusing the configurations for other devices of the same type.
- Ensure that when a new device is added, the same configuration is applied.

The management system can retrieve the current configuration from a device and apply the same to the new devices.

Simple Network Management Protocol (SNMP)

- SNMP is used at the application layer of the TCP/IP architecture.
- used to manage and monitor network and network faults.
- Also used to modify the configuration of the remote end devices of the network.
- The devices compatible with SNMP Protocol are modems, routers, switches, printers, servers, etc.



Simple Network Management Protocol (SNMP)

- **SNMP Components**

- 1. SNMP Manager

- · It is a centralized GUI-based node system that is used to monitor the network and is also called a Network Management System (NMS).
- · A manager is a host that runs the SNMP client program while the agent is a router that runs the SNMP server program.
- · Executes SNMP commands to monitor and configure the Managed Device.
- · It interfaces the bi-directional flow of information between the NMS node and the network elements.
- · Network elements are switches, routers, servers, modems, computer hosts, IP-based phone and video cameras, etc.

Simple Network Management Protocol (SNMP)

- 2. SNMP Agent

- · The agent is the module of network management software that is installed on a network device like a host PC, server and router, etc.
- · The agent is used to keep the information in a database while the manager is used to access the values in the database.
- · A server program on the agent checks the environment, if something goes wrong, the agent sends a warning message to the manager.
- · Agent software runs on the hardware or service being monitored, collecting data about disk space, bandwidth use, and other important network performance metrics. When queried by the SNMP manager, the agent sends the requested information back to the management system. An agent may also proactively notify the NMS if an error occurs.

Simple Network Management Protocol (SNMP)

- Management Information Database (MIB)
- This MIB database is a text file (.mib) that itemizes and describes all objects on a particular device that can be queried or controlled using SNMP.
- Each MIB item is assigned an object identifier (OID).
- Store all the information of the device attributes to be managed.

- Structure of MIB:
- · It is a group of information that comprises the variables that reside in the values relevant to the parameters of the network element in its stores. These variables are known as managed objects and are identified by an Object Identifier (OID).
- · MIB is a collection of object identifiers in a hierarchical format, and each can identify a variable that can be set or read by the SNMP.
- · The OIDs are of two kinds, scalar and tabular. The scalar one report only a single event instance means that the result is only one.
- · The Tabular object is a table that is a pool of all related OIDs and thus gives multiple results for one object value.

Simple Network Management Protocol (SNMP)

- SNMP Commands
- By deploying the SNMP, the network elements are managed by using three commands: Read, Write, and Trap.
- Read command is deployed by the NMS to monitor the managed network elements like routers, switches, etc. This action is completed by NMS by examining the various variables that are upheld by the network elements.
- Write command is deployed by the NMS to control the network elements. Through this command, the NMS can alter the values of the variables which are stored in the managed network elements.
- Trap command is utilized by the managed network elements to report the incidences and errors to the NMS.

Simple Network Management Protocol (SNMP)

- The SNMP request messages include the operations like ‘Get’, GetNext’, and ‘GetBulk’.

- · Get: By using this message, the NMS request to retrieve more than one variable from the SNMP agent.
- · GetNext: This operation permits the NMS to retrieve one or greater than one consequent variable from the SNMP agent.
- · GetBulk: This operation is correspondent to the consecutive GetNext operation. With this set of request messages, we can retrieve the database from the agent in bulk.
- · Response: It returns the variable data unit from the agent to the NMS in response to the Get and Set request PDUs.
- · Trap: This command is initiated by the SNMP agents. When an event occurs the agent sends a signal to the SNMP manager to acknowledge the occurrence in the form of this PDU.
- · InformRequest: Its function is the same as that of the Trap command. It includes the acknowledgment of receiving the packet from the SNMP manager.

Simple Network Management Protocol (SNMP)

- Limitations of SNMP

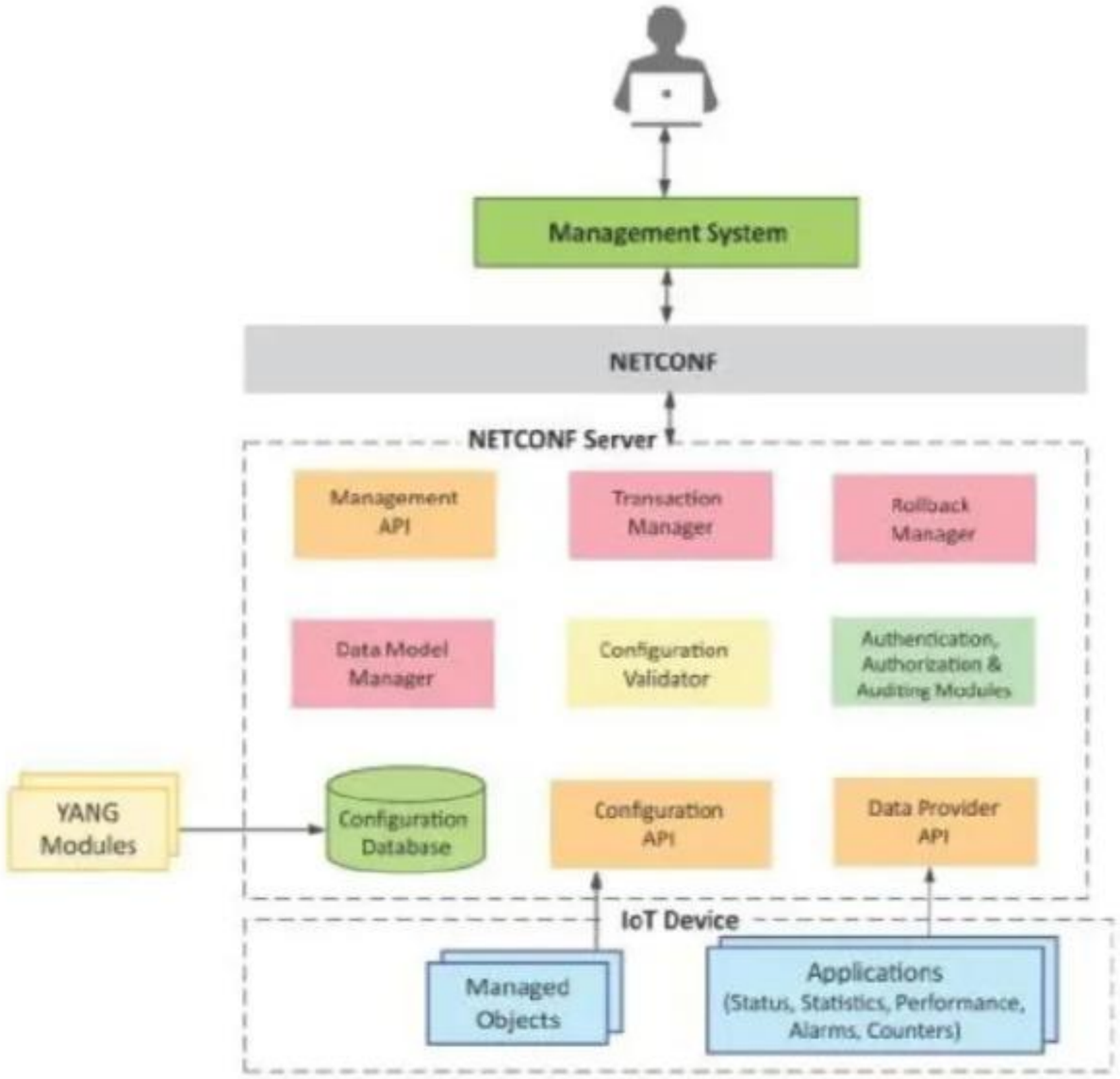
- Several limitations are.

- · SNMP is a connectionless protocol that uses UDP as the transport protocol, making it unreliable as there was no support for acknowledgment of requests.
- · MIBS often lack writable objects without which device configuration is not possible using SNMP.
- · With the absence of writable objects, SNMP can be used only for device monitoring and status polling.
- · It is difficult to differentiate between configuration and state data in mibs.
- · Retrieving the current configuration from a device can be difficult with SNMP.
- · SNMP does not support easy retrieval and playback of configurations.
- · Earlier versions of SNMP did not have strong security features making the management. Security features of SNMP, increased the complexity a lot.

Network Operator Requirements

- Ease of use
- Distinction between configuration and state data
- Fetch configuration and state data separately
- Configuration of the network as a whole
- Configuration transactions across devices
- Configuration deltas
- Dump and restore configurations Configuration validation
- Configuration database schemas
- Comparing configurations
- Role-based access control
- Consistency of access control lists
- Multiple configuration sets
- Support for both data-oriented & task oriented access control

IoT Systems Management with NETCONF-YANG



IoT Systems Management with NETCONF-YANG

- **Management System :** The operator uses a management system to send NETCONF messages to configure the IoT device and receives state information and notifications from the device as NETCONF messages.
- **Management API :** allows management application to start NETCONF sessions.
- **Transaction Manager:** executes all the NETCONF transactions and ensures that ACID properties hold true for the trasactions.

IoT Systems Management with NETCONF-YANG

- Rollback Manager : is responsible for generating all the transactions necessary to rollback a current configuration to its original state.
- Data Model Manager : Keeps track of all the YANG data models and the corresponding managed objects. Also keeps track of the applications which provide data for each part of a data model.
- Configuration Validator : checks if the resulting configuration after applying a transaction would be a valid configuration.

IoT Systems Management with NETCONF-YANG

Configuration Database : contains both configuration and operational data.

Configuration API : Using the configuration API the application on the IoT device can be read configuration data from the configuration datastore and write opeartional data to the opearational datastore.

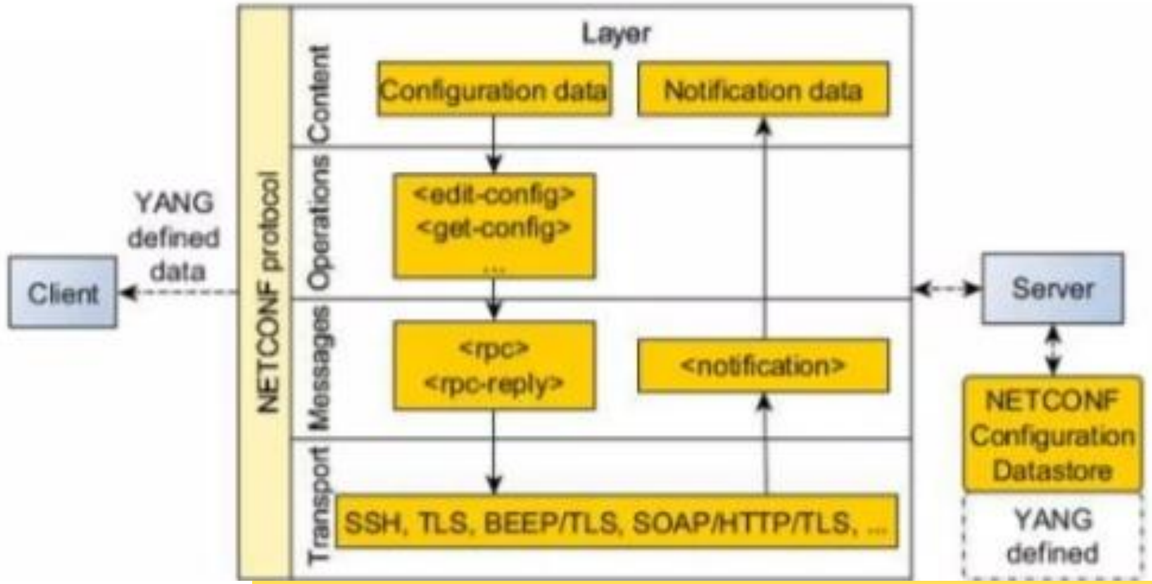
Data Provider API: Applications on the IoT device can register for callbacks for variousevents using the Data Provider API. Through the Data Provider API, the applications can report statistics and opeartional data.

IoT Systems Management with NETCONF-YANG

NETCONF

- Network Configuration Protocol (NETCONF) is a session-based network management protocol. NETCONF allows **retrieving state or configuration data** and **manipulating configuration** data on network device.
- NETCONF provides a clear separation of the **configuration and state data**.
- NETCONF uses XML-encoded Remote Procedure Calls (RPCs) for framing request and response message

NETCONF protocol layers.



IoT Systems Management with NETCONF-YANG

The NETCONF protocol is built on a four-layer approach :

1) Secure Transport Layer : Authentication and integrity can be provided by protocols such as TCP-based TLS and SSHv2.

2) Message Layer : A set of RPC messages and notifications are defined for use including <rpc>, <rpc-reply> and <rpc-error>.

3) Operations Layer : Defines a set of base protocol operations invoked by RPC methods using XML-encoding. These include <get-config>, <edit-config> and <get>.

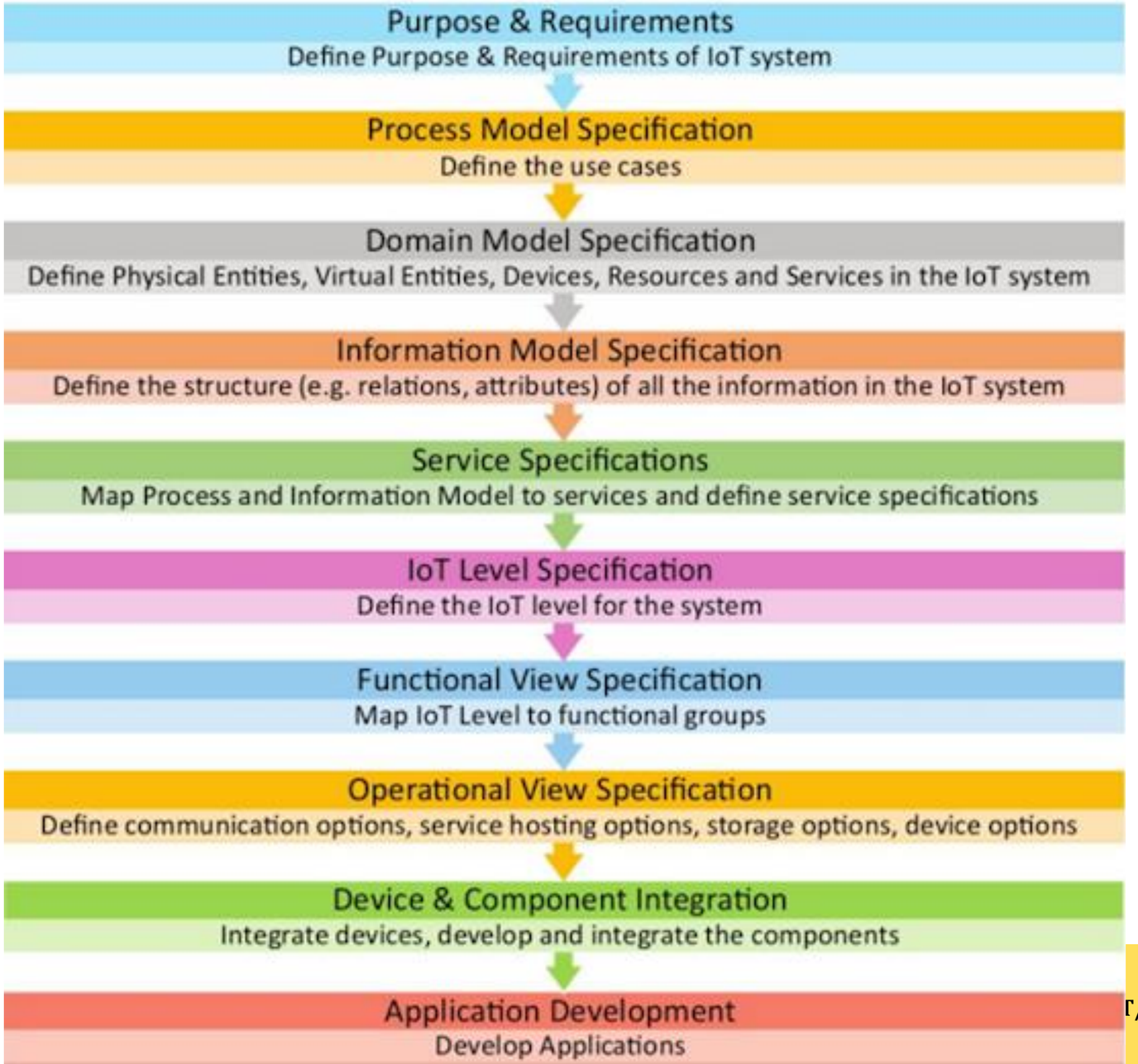
4) Content Layer : NETCONF data models and protocol operations use the YANG modeling language. A data model outlines the structure, semantics and syntax of the data

IoT Systems Management with NETCONF-YANG

Basic feature of NetConf.

1. Separates configuration and operation data
2. Client-server architecture.
3. Implemented using layered model._
4. Support multiple configuration data stores on managed devices.
5.. Configuration validation before commit.
6. Transactions over multiple managed devices

Iot Platform Design Methodology



lot Platform Design Methodology

- **Step 1: Purpose & Requirements Specification**

- In this step, design or prepare the system purpose, behavior, and requirements (such as data collection requirements, data analysis requirements, system management requirements, data privacy and security requirements, user interface requirements, ...) are captured.

- **Step 2: Process Specification**

- This step, formally described the purpose and requirement specifications of the IoT system.

- In a process diagram, the circle denotes the start of a process, the diamond denotes a decision box and the rectangle denotes a state or attribute (design flow chart).

Iot Platform Design Methodology

- **Step 3: Domain Model Specification**

- In this step, the IoT design methodology defines the Domain Model.
- The domain model describes the main concepts, entities, objects, attributes of the objects, and relationships between objects of the IoT system to be designed.
- The domain model provides an abstract view of the IoT domain, which is independent of any specific technology or platform.
- This domain model includes:

- . Physical Entity:
- A physical Entity is a discrete and identifiable entity in the physical environment (e.g. a room, a light, an appliance, a car, etc.).
- The IoT system provides information about the Physical Entity (using sensors) or performs actuation upon the Physical Entity (e.g., switching on a light).

- • Virtual Entity:
- A virtual Entity is a representation of the Physical Entity in the digital world.

lot Platform Design Methodology

- • Device:

- The device provides a medium for interactions between Physical Entities and Virtual Entities. Devices are either attached to Physical Entities or placed near Physical Entities.
- Devices are used to gather information about Physical Entities (e.g., from sensors), perform actuation upon Physical Entities (e.g. using actuators), or used to identify Physical Entities (e.g., using tags).

- • Resource: -
- Resources are software components that can be either "on-device" or "network resources".
- On-device resources are hosted on the device and include software components that either provide information on or enable actuation upon the Physical Entity to which the device is attached.
- Network resources include the software components that are available in the network (as a database).

- • Service
- Services provide an interface for interacting with the Physical Entity.
- Services access the resources hosted on the device or the network resources to obtain information about the Physical Entity or perform actuation upon the Physical Entity.

lot Platform Design Methodology

- **Step 4: Information Model Specification**
- This step defines Information Model.
- The Information Model defines the structure of all the information in the IoT system.
- example, attributes of Virtual Entities, relations, etc.
- The information model does not describe “how the information is represented or stored”.
- The information model defines the list of the Virtual Entities, their attributes, and the relations of the domain model.

Iot Platform Design Methodology

- **Step 5: Service Specifications**
- In this step, Service specifications define the services in the IoT system such as service types, service inputs/output, service endpoints, service schedules, service preconditions, and service effects.
- These services either change the state or attribute values or retrieve the current values.
- The Mode service is a RESTful web service that sets the mode to auto or manual (PUT request), or retrieves the current mode (GET request).
- The mode is updated to/retrieved from the database.
- The State service is a RESTful web service that sets the light appliance state to on/off (PUT request) or retrieves the current light state (GET request).
- The state is updated to/retrieved from the status database.
- The Controller service runs as a native service on the device.

Iot Platform Design Methodology

- **Step 6: IoT Level Specification**

- In this step define the IoT level for the system. Five types of IoT deployment levels are used according to different conditions.

- IoT Level 1
- • This level consists of an air conditioner, temperature sensor, data collection and analysis, and control & monitoring app.
- • The data sensed is stored locally.
- • All the control actions are performed through the internet.

- IoT Level 2
- • This level consists of an air conditioner, temperature sensor, Big data (Bigger than level -1, data analysis done here), cloud, and control & monitoring app.
- • This level-2 is complex and the rate of sensing is faster compared to level-1.

Iot Platform Design Methodology

- IoT Level 3
- • this level consists of an air conditioner, temperature sensor, big data collection (Bigger than level-1), cloud (for data analysis), and control & monitoring app.
- • Data here is voluminous i.e. big data. The frequency of data sensing is fast and collected sensed data is stored in the cloud as it is big.

- IoT Level 4
- • This level consists of multiple sensors, data collection and analysis, and a control & monitoring app.
- • Data uploaded by sensors to the cloud separately.

- IoT Level 5
- • This level consists of multiple sensors, coordinator node, data collection and analysis, and control & monitoring app.
- • For huge data using multiple sensors at a faster rate and simultaneously.
- • The data collection and data analysis is performed at the cloud level using the mobile app or web app.

Iot Platform Design Methodology

- **Step 7: Functional View Specification**

- In the seventh step define the Functional View.
- The Functional View (FV) defines the functions of the loT systems grouped into various Functional Groups (FGs).
- Each Functional Group either provides functionalities for interacting with instances of the Domain Model.

- A Functional View includes:
- · Device:
  - The device FG contains devices for monitoring and control.
- · Communication:
  - The communication FG handles the communication protocols and APIs (such as REST and WebSocket) that are used by the services and applications to exchange data over the network. for the IoT system.
  - These are the backbone of IoT systems and enable network connectivity.

lot Platform Design Methodology

- Services:
  - The service FG includes various services involved in the IoT system such as services for device monitoring, device control services, data publishing services, and services for device discovery.
- Management:
  - The management FG includes all functionalities that are needed to configure and manage the loT system.
- Security:
  - The security FG includes security mechanisms for the loT system such as authentication, authorization, data security, etc.
- Application:
  - The application FG includes applications that provide an interface for the users to control and monitor various aspects of the loT system.
  - Applications also allow users to view the system status and the processed data.

Iot Platform Design Methodology

- **Step 8: Operational View Specification**
- In this step define the Operational View Specifications. IoT system deployment and operation are defined, such as service hosting options, storage options, device options, application hosting options, etc.

- • Devices:
- The computing device (Raspberry Pi), light-dependent resistor (sensor), relay switch (actuator). . Communication APIS: REST APIs

- • Communication Protocols:
- Link Layer - 802.11. Network Layer-IPv4/IPv6, Transport -TCP, Application - HTTP.

lot Platform Design Methodology

- • Services:
- 1. Controller Service - Hosted on the device, implemented in Python, and run as a native service.
- 2. Mode service - REST-ful web service, hosted on a device, implemented with Django-REST Framework.
- 3. State service - REST-ful web service, hosted on a device, implemented with Django-REST Framework.

- Application:
- Web Application - Django Web Application, Application Server - Django App Server, Database Server - MySQL.

- • Security:
- Authentication: Web App, Database

- • Management:
- Application Management - Django App Management
- Database Management - MySQL DB Management, Device Management - Raspberry Pi device Management.

lot Platform Design Methodology

- **Step 9: Device & Component Integration**

- In this step integration of the devices and components design such as minicomputer, LDR sensor, and relay switch actuator.

- **Step 10: Application Development**

- The final step in the IoT design methodology.
- It is to develop the IoT application.
- The application has controls for the mode (auto-on or auto-off) and the light (on or off).