**Internet of Things**

The network of physical objects is called Internet of Things (IoT). Things, People and Cloud Services get connected via Internet to enable new use cases and business models. Improved VLSI Technology for Miniaturization and MEMS Technology for Sensing both leverage IoT. Widespread adoption of intellectual properties is accomplished through it.It is utilized to calculate economics. It serves descriptive, diagnostic, prescriptive, and predictive purposes in data analytics. Its connectivity is faster.The growth of cloud computing gives big data

Scalability.

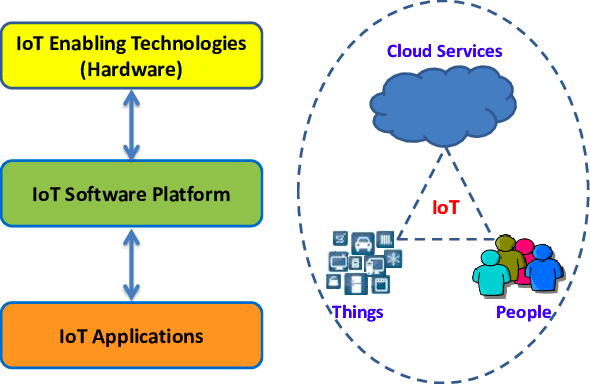


Figure 1: Internet of Things

**History of IoT:**

First IoT Device : RFID (1940-1950), Major efforts in development for Tracking and identifying aircrafts in World War II ( Friends or Foe )

1960 : RFID was used for monitoring nuclear and other hazardous materials, RFID companies founded

1973 : Mario W. Cadullo received the first US Patent for an active RFID tag.

1980 : RFID research started, and it marks the beginning of transforming RFID into more widespread technology

1990 : First UHF Reader was invented, RFID usage expanded to shipments, Walmart introduces their RFID program



Figure 2: RFID

Progression in 1980’s : Cloud and Server Space ( Data moved to centralized server )

Progression in 1990’s : Machine to Machine interaction

1995 : First cellular module built, First GPS network ( version 1 ) complete 1998 : IPv6 adds 2^128 new IP addresses

1999 : Kevin Ashton of MIT coins a new term IoT

Progression in 2000-2010 : Fog oriented architectures ( Central Server to Regional Server located closer to Data Server subnetwork )

2000 : LG announces first smart fridge 2007 : First iPhone released

2008 : First International Conference on IoT held.

2009 : Google started testing self driving cars.

Progression in 2010-onwards : High Processing power and Edge computing

2013 : Google glass is released

2014 : Amazon releases Echo ( smart home market opens ) 2015 : GM, Uber, Tesla are testing self driving cars

2017- : IoT continues to grow,

**Physical and Virtual Things**

Physical things exist in the physical world and are capable of being sensed, actuated and connected.

**Ex:** electrical and mechanical equipments; goods; robots etc.

Virtual things exist in the information world and are capable of being stored, processed and accessed.

**Ex:** multimedia content and application software.

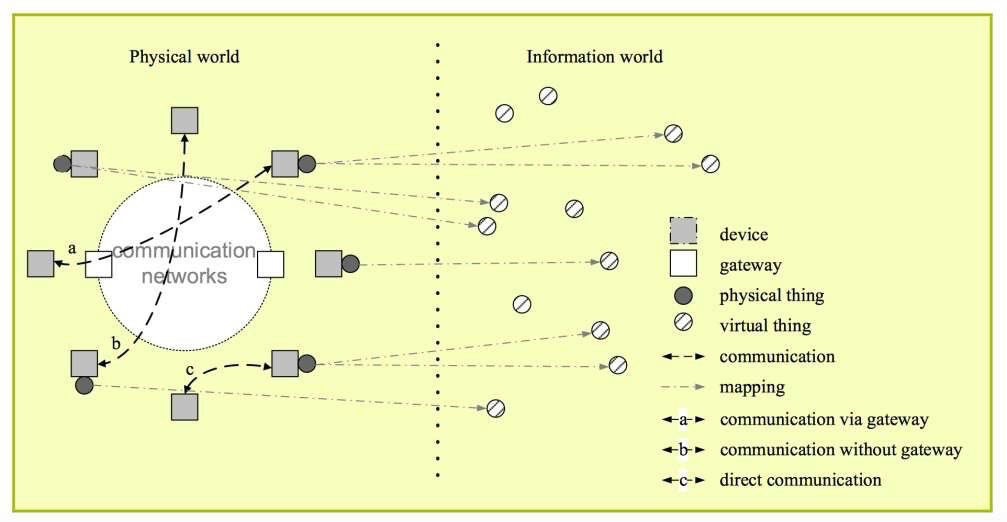


Figure 3: Physical and virtual world

**Connectivity Considerations:**

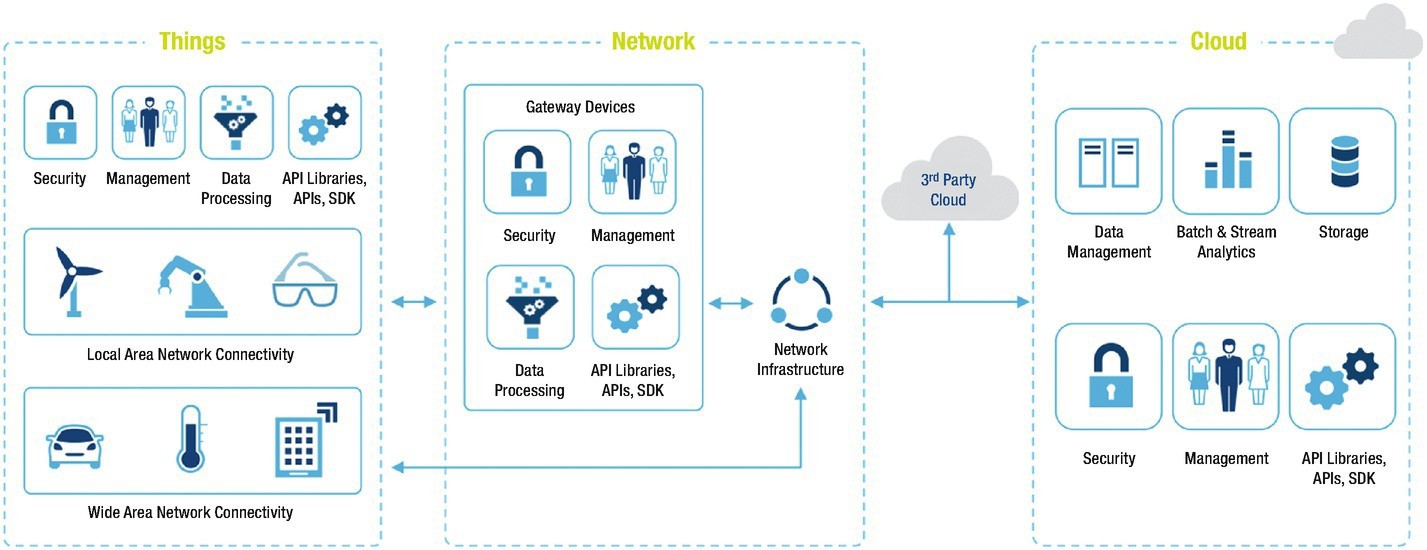
Network architecture is determined by how IoT devices are connected to the outside world. Hardware specifications and pricing for IoT devices are determined by the communication method chosen.A single networking paradigm is insufficient to meet all of the needs of the consumer or IoT device due to the presence of numerous applications for IoT enabled devices.Network complexity includes device interference, network management, network heterogeneity, and internal network protocol standardization.

Figure 4: Connectivity considerations

**Network considerations:**

**LAN :** It is a Local short range communication, may or may not connect to Internet.

**WAN** : it is a connection of various network segments, connects to internet.

**Node** : It connects to other nodes via LAN, maybe connected to other nodes via WAN directly

**Gateway** : A router connecting LAN to WAN and Internet, can implement several LAN and WAN, forwards packets between LAN and WAN

**Proxy** : It performs active application layer functions between nodes and other entities

**Logical Design of IoT Based Systems**

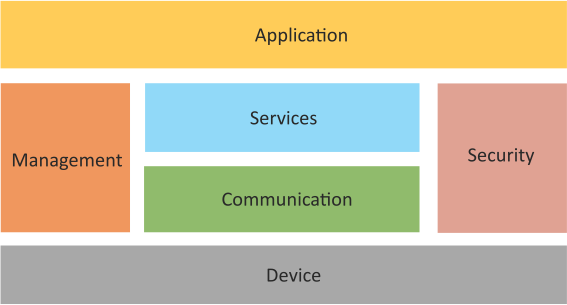


Figure 5: Logical Design of IoT

Without delving into the intricate details of the implementation, logical design of an IoT system refers to an abstract representation of the entities and processes.

An Internet of Things (IoT) system is made up of many functional building parts that enable identification, sensing, actuation, communication, and management.An IoT system comprises the following components :

**Device :** Identification, remote sensing, actuation, and remote monitoring are all possible with an IoT device.

**Resources:** Resources are software components on the IoT device that are used to access, process, and store sensor data as well as to control actuators that are connected to it. The software parts that give the device network connectivity are also considered resources.

**Controller Service:** A native service called the controller service runs on the device and communicates with the web services. The controller service transmits information from the device to the web service and gets instructions for managing the device from the application (through web services).

**Database** : The data produced by the IoT device is stored in databases, which may be local or on the cloud.

**Web Service :** The IoT device, application, database, and analysis components are connected by web services. Web services can be implemented using the WebSocket protocol (WebSocket service) or utilizing HTTP and the REST principles (REST service).

**Analysis Component :** This is in charge of processing the IoT data and producing information that is simple for the user to grasp.

**Application :** IoT applications offer a user interface so that users may manage and see various functions of the IoT system. Users can also monitor the system status and the data that has been processed using applications.

**IoT Levels and Deployment**

**IoT Level – 1:**

IoT Level 1 uses a single node to host applications and carry out sensing and/or actuation.

At the node, data is kept. The node is where the application runs. They are appropriate for modeling inexpensive solutions where the data is small and the processing demands are modest.

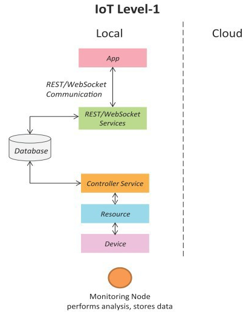


Figure 6: IoT Level 1

**Application:** Smart home

Sensors used : Light, Temperature, Motion, Camera



Figure 7: Smart Home

**IoT Level – 2**

IoT Level 2 uses a single node for local analysis, sensing, and/or actuation.

Applications are cloud-based, and data is saved there as well.They work well for modeling when there is a lot of data. The criteria for analysis, however, are not computationally demanding and can be completed locally.

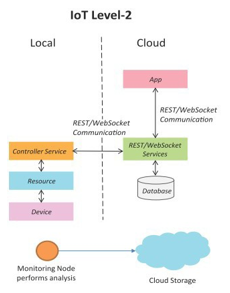


Figure 8: IoT Level 2

**Application:** Smart irrigation

Sensors used : Temperature, Humidity, Pressure



Figure 9: Smart irrigation

**IoT Level - 3**

IoT Level 3 has a single node that handles local analysis, sensing, and/or actuation.

Applications are cloud-based, and data is saved there as well.They work well for modeling when there is a lot of data. However, analysis requirements require a lot of processing power.

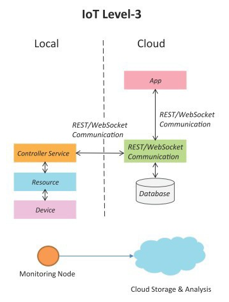


Figure 10: Level 3

**Application:** Tracking Package Delivery

Sensors used : Gyroscope and Accelerometer



Figure 11: Package Delivery

**IoT Level – 4**

Multiple nodes at IoT Level 4 do local analysis. Applications are cloud-based, and data is saved there as well. Local and cloud-based observer nodes with the ability to subscribe to and receive cloud-based data.They are appropriate in situations requiring several nodes, large amounts of data, and computationally demanding analysis needs.

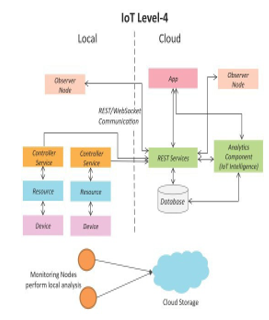


Figure 12: IoT Level 4

**Application:** Noise Monitoring System

Sensors used : Sound Sensors



Figure 13: Noise Monitoring System

**IoT Level - 5**

Multiple end nodes and one coordinator node make up IoT Level 5.The end node performs actuation and sensing.Data from end nodes is gathered by collector nodes and sent to the cloud.Applications are cloud-based, and data is saved there as well.

For wireless sensor network-based solutions with large amounts of data and computationally demanding analysis needs, Level 5 IoT systems are appropriate.

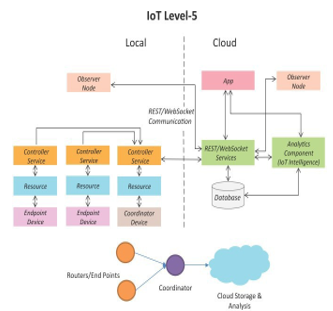


Figure 14: IoT Level 5

**Application:** Forest Fire Detection System

Sensors used : temperature, smoke, weather, slope of earth, wind speed, speed of fire, flame length



Figure 15: Forest Fire

**IoT Level - 6**

Multiple autonomous end nodes at IoT Level 6 execute sensing and/or actuation and transfer data to the cloud. Applications are cloud-based, and data is saved there as well.

Data is analyzed by the analytics component and stored in a cloud database. With the cloud-based application, the outcomes are visualized. All end nodes' status is known to the centralized controller, which also delivers control instructions to the nodes.

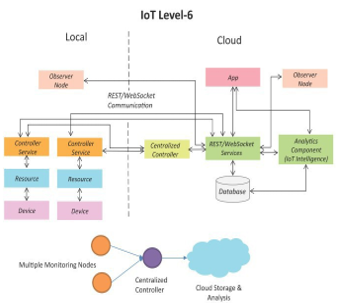


Figure 16: IoT Level 6

**Application:** Weather Monitoring Station

Sensors used : wind speed direction, solar radiation, temperature, relative humidity, precipitation, snow depth, barometric pressure,soil moisture



Figure 17: Weather Monitoring

**Communication Models : Request – Response**

In the request-response communication model, the client makes requests of the server, which the server then responds to.The response is given to the client when the server selects how to respond, acquires the data, gets the resource representations, and prepares it.

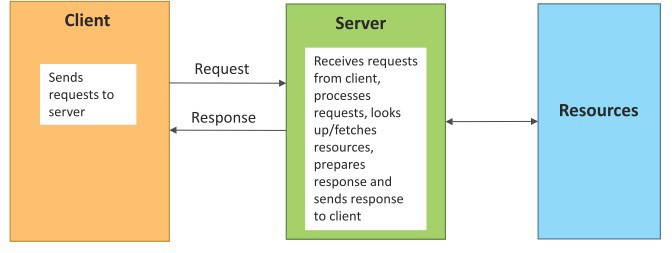


Figure 18: Request- Response Communication models

**Communication Models : Publish- Subscribe**

Publishers, brokers, and customers all participate in the publish-subscribe communication mechanism.The statistics are sourced from publishers. Publishers send information to the broker-managed topics. The consumers are unknown to publishers.Customers subscribe to the topics that the broker manages.Data for a topic is sent to all subscribers by the broker

once it has been received from the publisher.

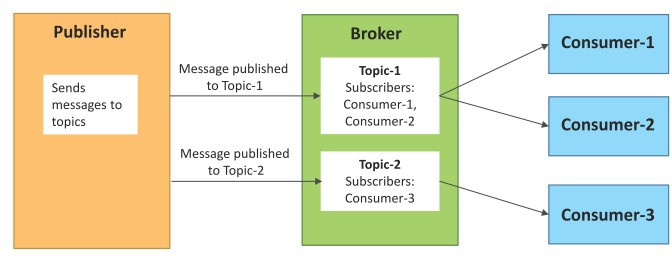


Figure 19: Publish Subscribe communication models

**Communication Models : Push – Pull**

The data producers push the data into queues, while the consumers pull the data from the queues, in the push-pull communication model.The consumers do not need to be known by the producers. The messaging between producers and consumers can be separated with the use of queues.Additionally, queues function as a buffer, which is beneficial when there is a discrepancy between the rates at which data is produced and consumed.

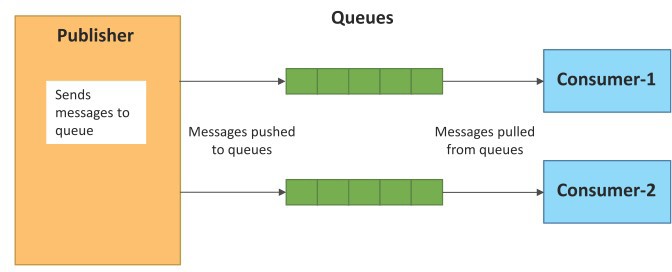


Figure 20: Push-Pull communication models

**Communication Models : Exclusive Pair**

Exclusive Pair is a persistent connection between the client and the server that uses a bidirectional, fully duplex communication architecture. Once established, a connection does not need to be closed unless the client requests it. After establishing a connection, the client and server can communicate by sending messages.

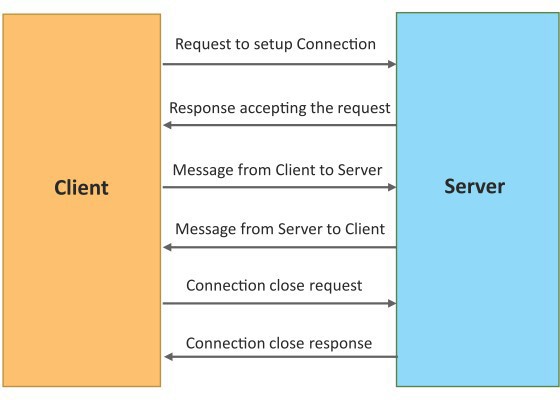


Figure 21: Exclusive pair.

**REST based APIs**

Web services and web APIs that concentrate on a system's resources and how resource states are addressed and communicated can be designed using a set of architectural principles called Representational State Transfer (REST).

REST APIs follow the request– response communication model.

REST architectural constraints apply to the components, connectors and data elements within a distributed hypermedia system

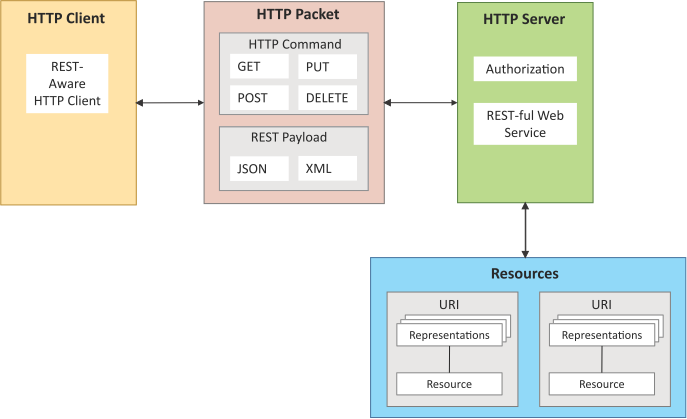


Figure 22: REST based API

**Machine to Machine (M2M) Services**

Machine to Machine refers to the networking of equipment for the exchange of data as well as remote monitoring and control.M2M services are solutions that concentrate on remote data gathering and transfer from embedded sensors installed on remote fixed or mobile assets.Using M2M technology, smart mobile devices are utilized to communicate with and manage numerous equipment (devices).

Machines (or M2M nodes) that contain embedded hardware modules for sensing, actuation, and communication make up an M2M area network.In M2M area networks, Zigbee, Bluetooth, 6LoWPAN, and IEEE.802.15.4 are employed as communication technologies. They employ non-IP or proprietary protocols.Access to the M2M area network is provided by the communication network. IP-based connectivity protocols are used in communication networks.

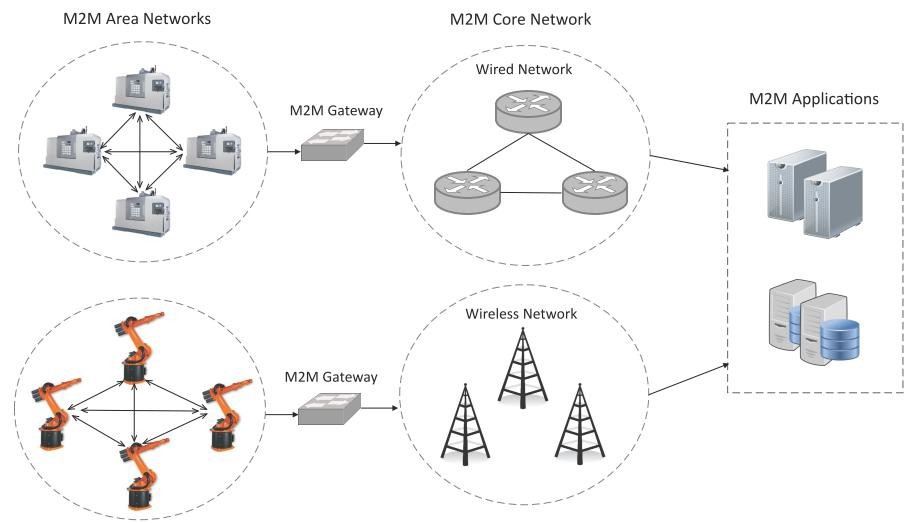


Figure 23: M2M

**Machine to Machine (M2M) Gateway**

The M2M nodes within one network are unable to connect with nodes in an external network because M2M area networks use non-IP based protocols.M2M gateways are used to allow communication between distant M2M area networks.

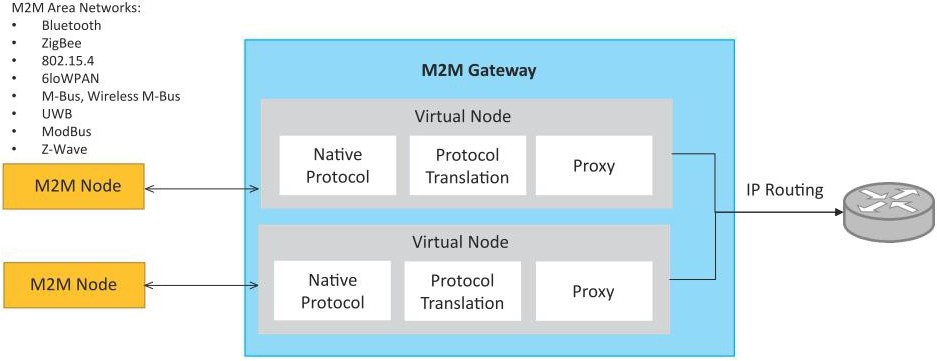


Figure 24: M2M Gateway

**Software Defined Networking**

Networking architecture that centralizes the network controller (which has data contained in it) and divides the control plane (network control) and data plane (network devices). Separation of control makes it possible to separate network services from the underlying parts and aids in treating the network as a logical entity.SDN controllers simplify configuration, maintenance, and provisioning while maintaining a uniform view of the network.Instead of specialized hardware, underlying infrastructure makes use of packet forwarding devices in traditional networks.

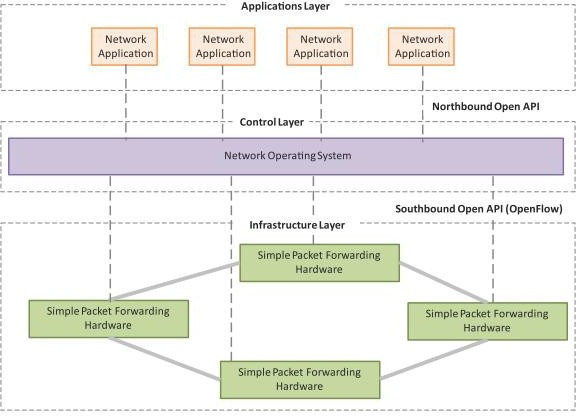
.

Figure 25: Software Defined Networking

A plane is an abstract conception of where certain processes take place.

Control plane : Control plane shows how packets are forwarded; Ex: process of creating routing tables.

Data plane : Data plane shows how packets should forwarded; Ex: access control, traffic monitoring.

Management plane : The management plane collect, measure and configure equipments.

**SDN Controller**

An application for enabling intelligent networking by managing flow control.They are built upon protocols like Open flow, which let servers instruct switches on where to send packets.

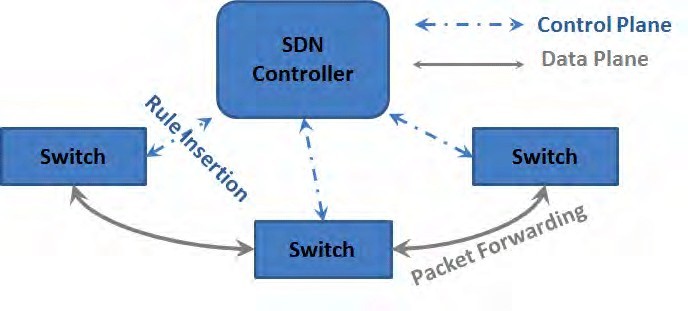


Figure 26: SDN Controller

**Network Function Virtualization ( NFV )**

Network Function Virtualization (NFV) is a technology that leverages virtualization to consolidate the heterogeneous network devices onto industry standard high volume servers, switches and storage.

Example : Virtual firewalls, load balancers, WAN accelerators, intrusion detection services

NFV can provide the infrastructure on which SDN can run.

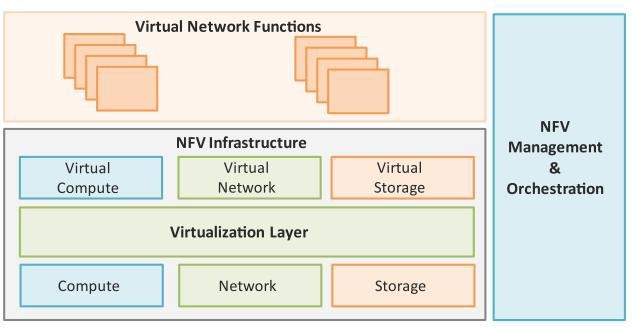


Figure 27: Network Function Virtualization

* Virtualized Network Function (VNF) : VNF is a software implementation of a network function which is capable of running over the NFV Infrastructure (NFVI)
* NFV Infrastructure (NFVI) : NFVI includes compute, network and storage resources that are virtualized.
* NFV Management and Orchestration : NFV Management and Orchestration focuses on all virtualization-specific management tasks and covers the orchestration and life-cycle management of physical and/or software resources that support the infrastructure virtualization, and the life-cycle management of VNFs.

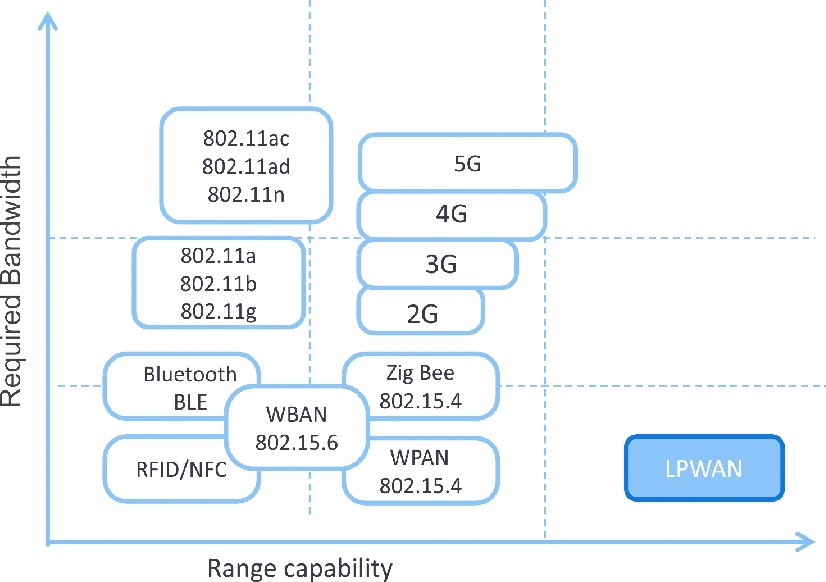


Figure 28: Bandwidth Requirements