BITS Pilani
Pilani Campus

BITS Pilani presentation

Paramananda Barik
CS&IS Department

innovate achieve lead

BITS Pilani
Pilani Campus

SEZG586/SSZG586, Edge Computing

Lecture No.7

Case Study: EdgeOS$_H$

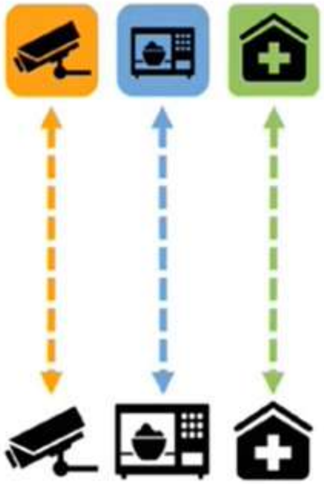
A Home Operating System for Internet of Everything

Smart home design

The lack of a home operating system makes it very difficult to manage devices, data, and services.

EdgeOS$_H$

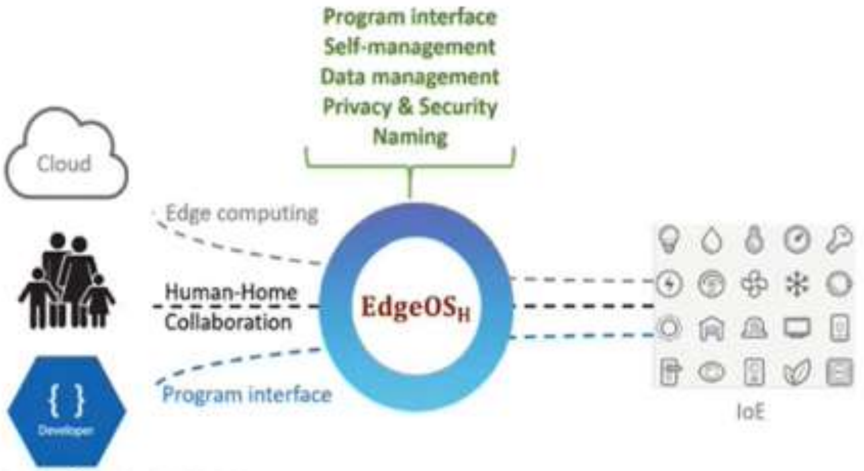
Systems work in a silo-based manner and cannot be connected or communicate with other systems



EdgeOS$_H$ Overview and Design

Benefits

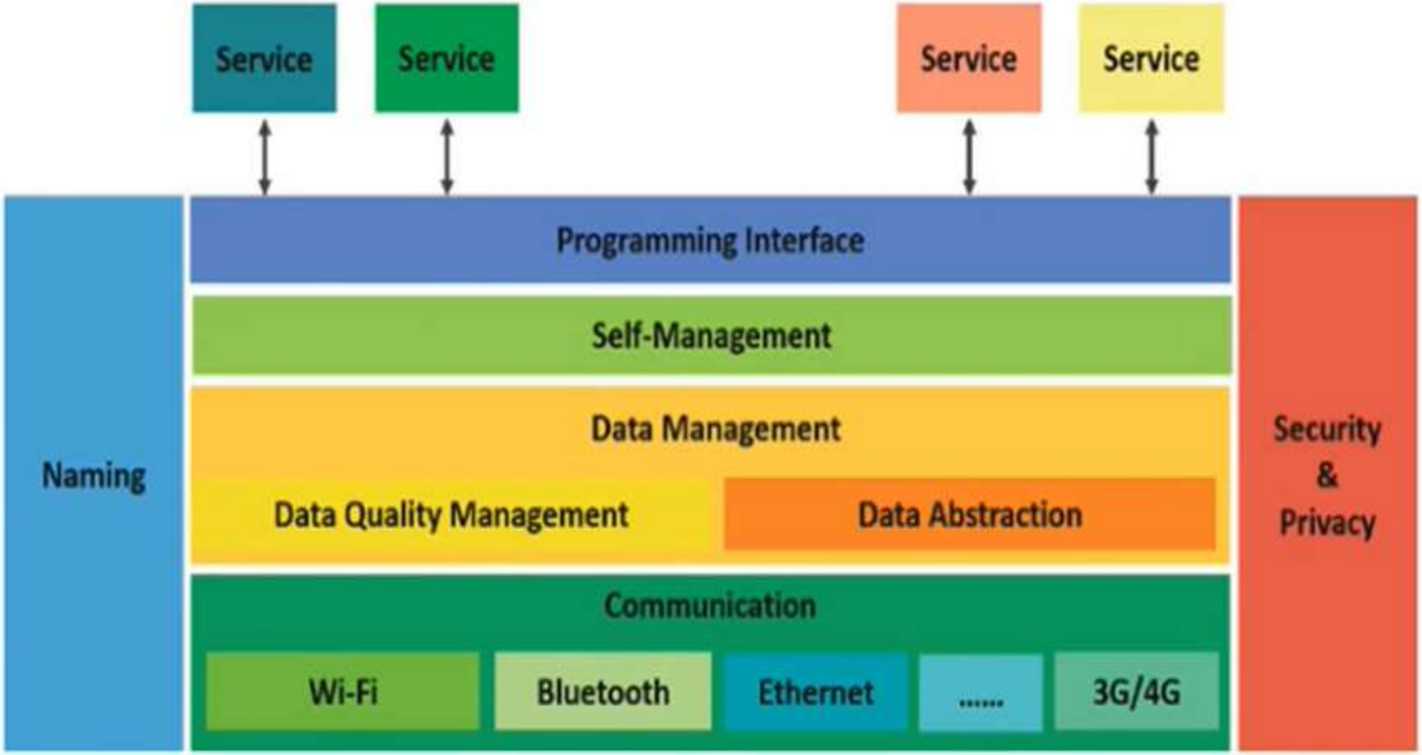
- Network load could be reduced
- Service response time could be decreased
- Data is better protected



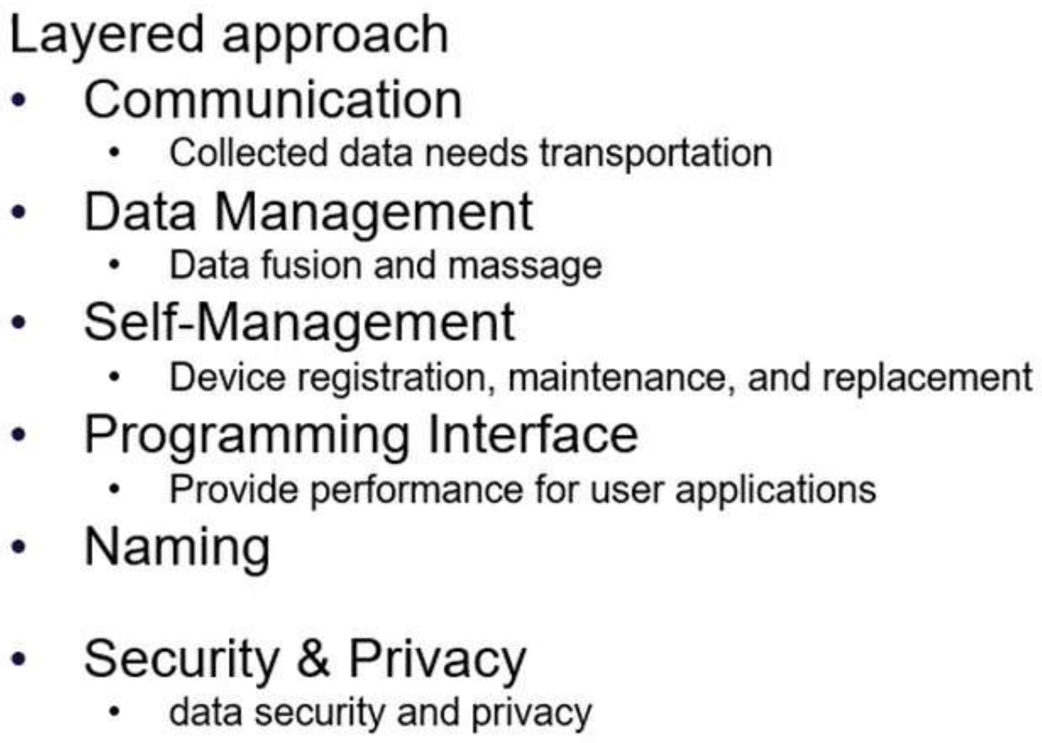
Challenge

- Home environment is very dynamic - hardware provided by different manufacturers

Overview


Service
Service
Service
Service
Programming Interface
Self-Management
Data Management
Data Quality Management
Data Abstraction
Communication
Wi-Fi
Bluetooth
Ethernet
......
3G/4G
Naming
Security
&
Privacy

Overview



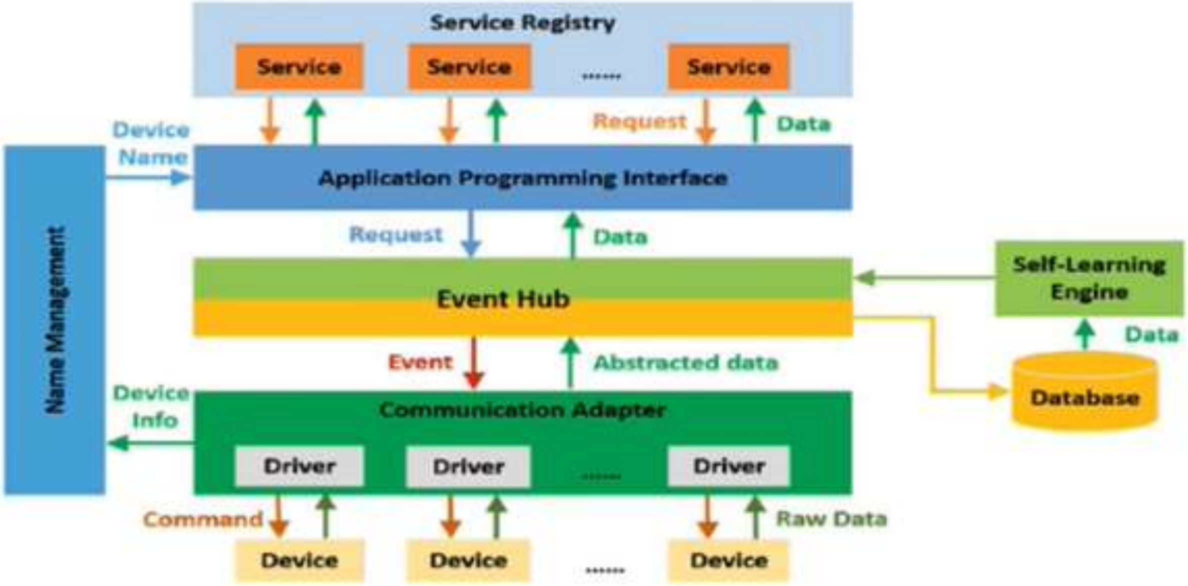
Layered approach

- Communication
  - Collected data needs transportation
- Data Management
  - Data fusion and massage
- Self-Management
  - Device registration, maintenance, and replacement
- Programming Interface
  - Provide performance for user applications
- Naming
- Security & Privacy
  - data security and privacy

Design

The design consists of 7 components:

- Communication Adapter, Event Hub, Database, Self-Learning Engine
- Application Programming Interface, Service Registry, Name Management



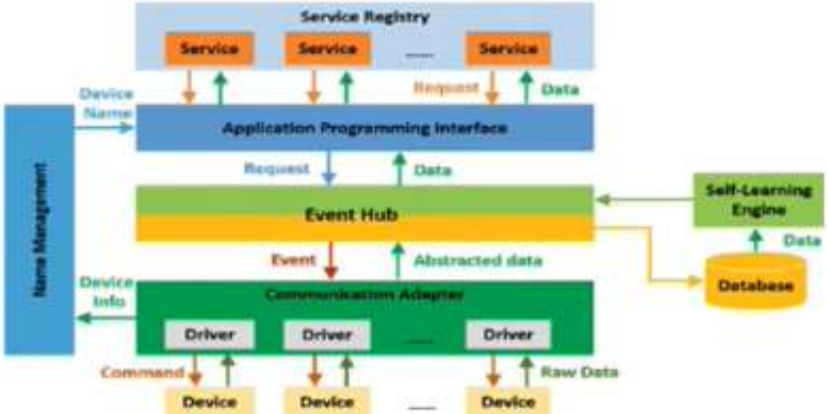
7 Components

Communication Adapter gets access to devices by the embedded drivers

- Drivers - responsible for sending commands to devices and collecting state data
- Combines different communication methods and provide a uniform interface for upper layers

Event Hub maps two layers in the logical view:

- Data Management and Self-Management layers
- Responsible for capturing system events and sending instructions to lower levels



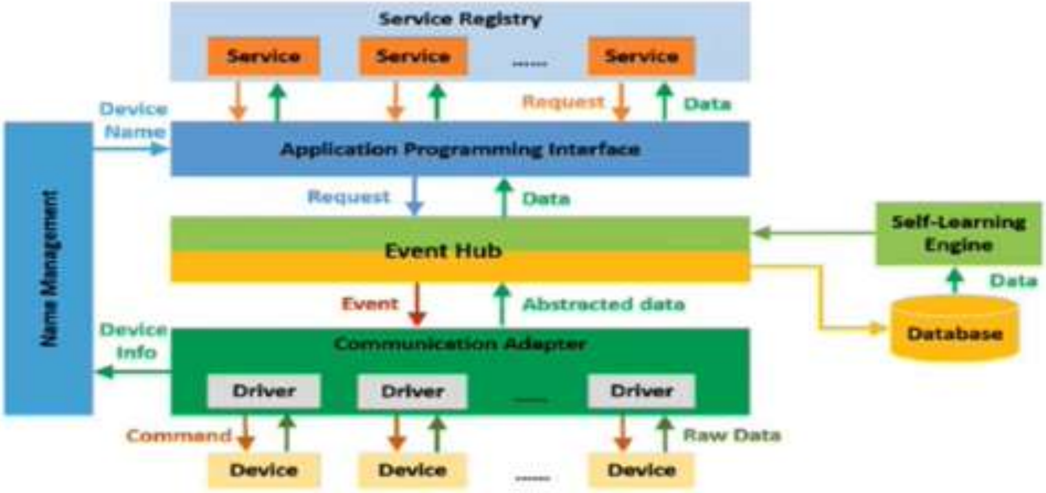
7 Components

Database is another layer in Data management layer

- Event Hub stores data in the Database
- Stored data is utilized by the Self-Learning Engine

Self-Learning Engine

- creates a learning model
- analyze user behavior, generate the personal model for the user



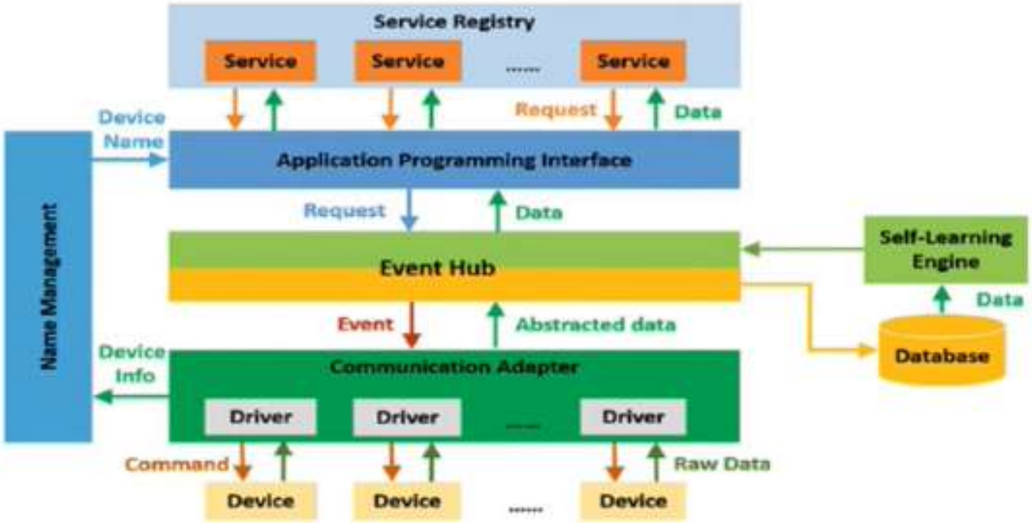
7 Components

Application Programming Interface (API) and Service Registry

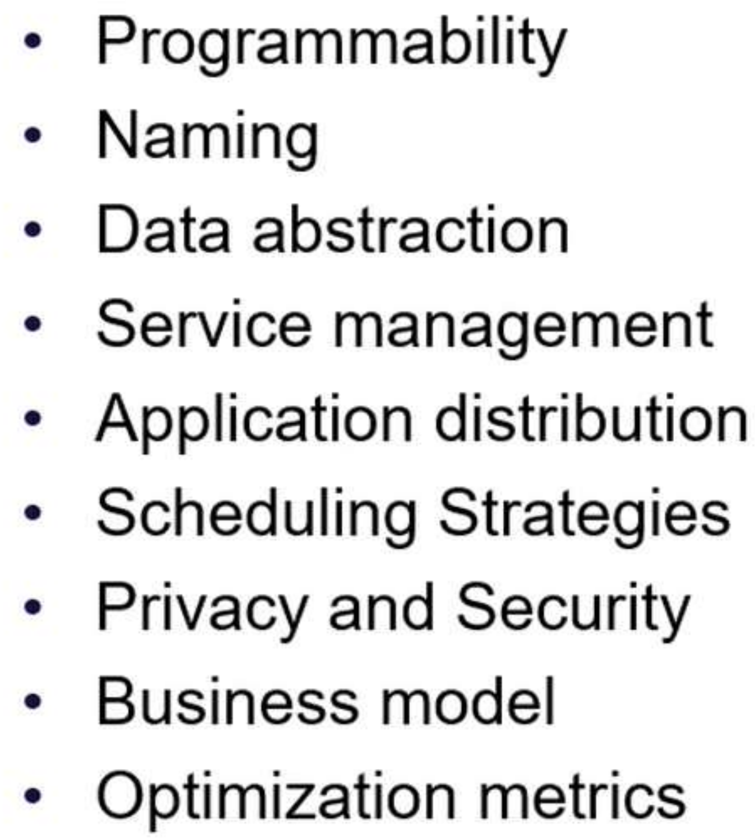
- located in the upper layers of the system and are utilized for third-party services.

Name Management

- Helps the system keep devices organized
- A name for it using the following rule: location (where), role (who), and data description (what)



Challenges and Opportunities of Edge Computing



- Programmability
- Naming
- Data abstraction
- Service management
- Application distribution
- Scheduling Strategies
- Privacy and Security
- Business model
- Optimization metrics

Programmability

Programming on Cloud

Users program and deploy the code on the cloud

Who decides, where is it computed?

Program written in

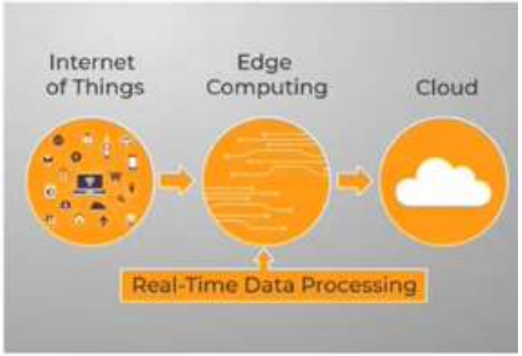
- One programming language
- Compiled for a certain target platform
- Runs in the cloud

What happens in Edge?

Computation is offloaded from the cloud or ?? Devices

Edge nodes are most likely ?? Platforms

Runtime on these nodes might differ



What happens in Edge?

How is this addressed?

Computing Stream - a serial of functions/computing applied to the data

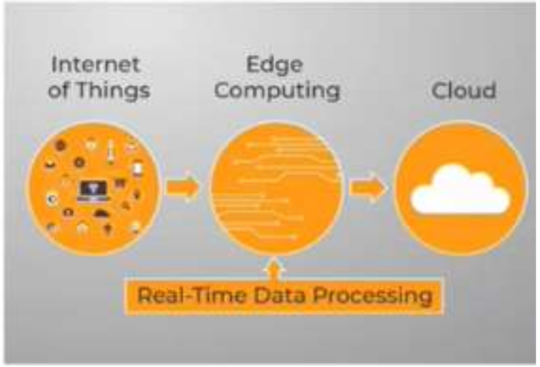
The function can be reallocated

- The data and state along with the function should also be reallocated

Software-defined computing flow

Data Processed

- Data generating devices
- Edge nodes
- Cloud environment



Example -1

ESP32 module with an integrated camera

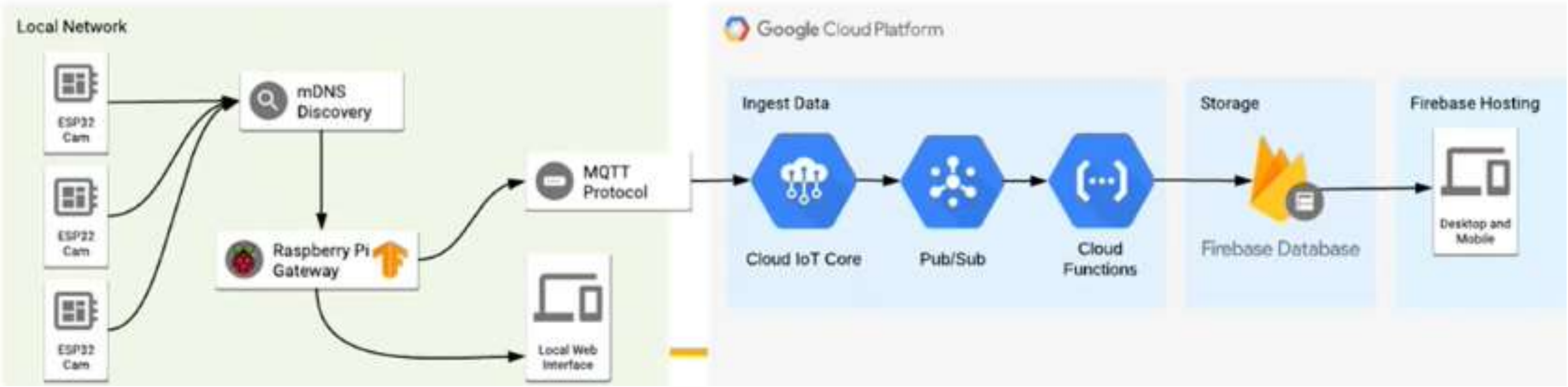
Raspberry Pi board as a local server

image classification using **Tensorflow**

Classified data is sent to the cloud securely using **Cloud IoT Core**

Data processed using **Firebase Cloud Functions**,

Data stored on **Firebase**



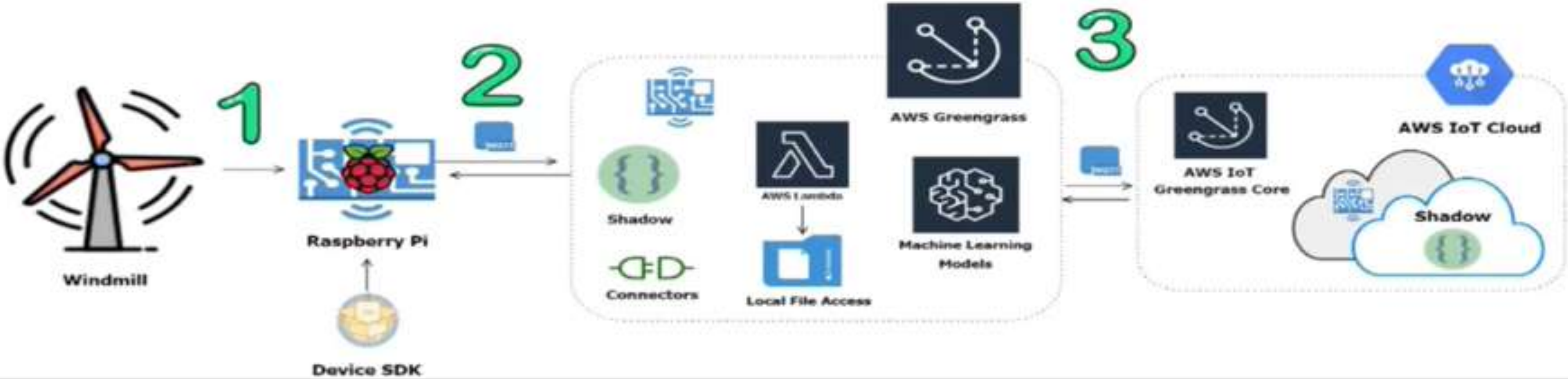
Example 2 – AWS IOT Greengrass for Windmail monitoring

Local Greengrass Core Device: Sensor data (temperature, humidity and velocity) is collected and published using MQTT protocol.

A Lambda function which is running on Greengrass Core subscribes to that topic payload (Sensors Data)

Data stored it on local file storage

As data is being collected, it is analyzed for future Predictive Maintenance



Example 3: Face Recognition Model at the Edge with AWS IoT Greengrass

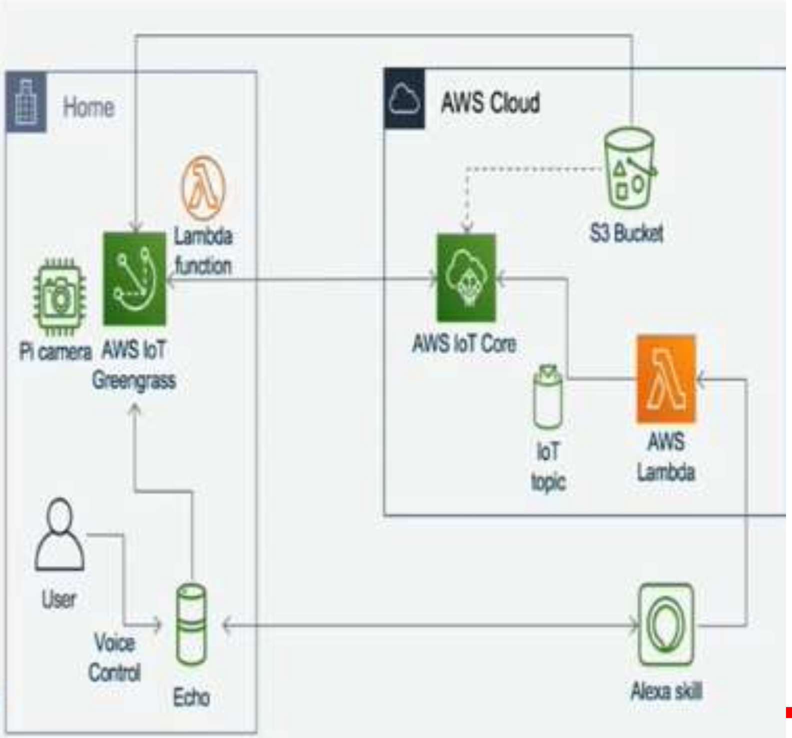
The facial recognition model and datasets uploaded to an Amazon S3 bucket

Used to create AWS Lambda function for recognition

AWS IoT Greengrass synchronizes the required files to the Raspberry Pi.

Echo Dot runs as a trigger. When Echo Dot listens to a command such as,"Alexa, open Monitor," it calls an Alexa skill to send a message to AWS IoT Core.

AWS IoT Core invokes the recognition Lambda function, which is deployed on Raspberry Pi local storage, and if the Lambda function recognizes the identity of the guest, the door opens.



Naming

Why is naming of things important?

- Addressing
- Things identification
- Data communication
- Programming

The naming scheme for Edge computing needs to handle:

- Mobility of things
- Highly dynamic network topology
- Privacy and security protection
- Scale

Naming : Edge OS

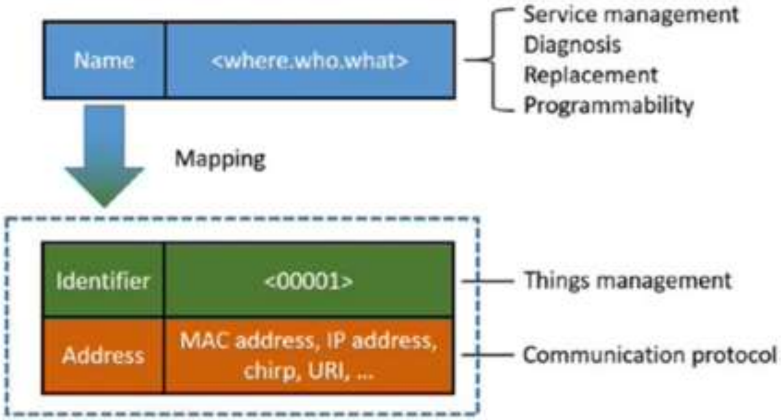
Naming mechanism in EdgeOS:

EdgeOS assign the network address to each thing

Human-friendly name which describes the following information: location (where), role (who), and data description (what)

Example: kitchen.oven2.temperature3

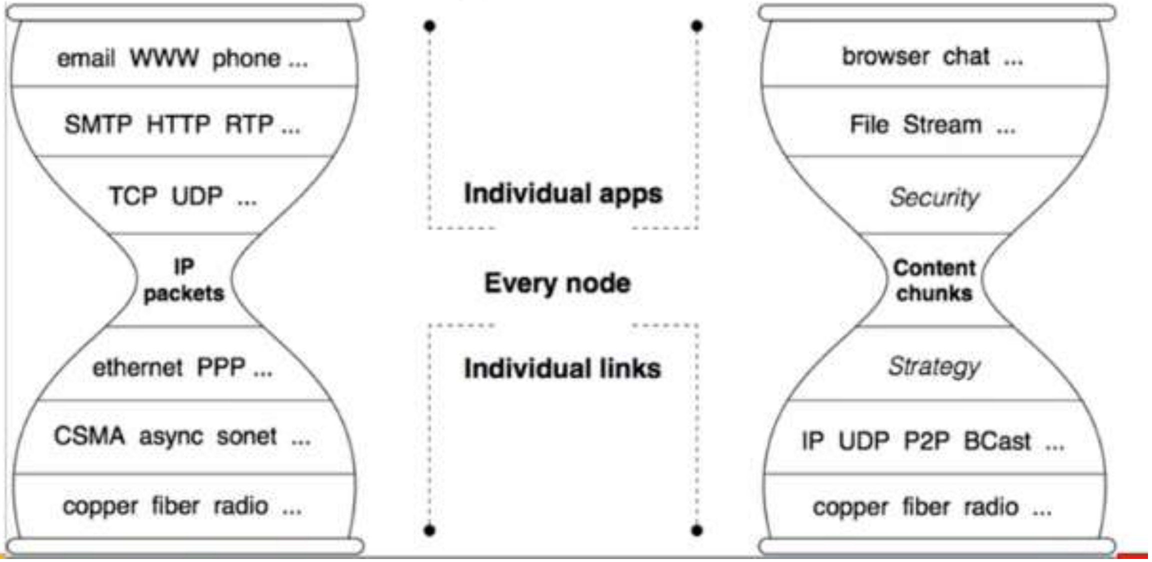
EdgeOS will assign identifier and network address to this thing



New naming mechanism

Named Data Networking

- NDN - hierarchically structured name for content/data-centric network
- Extra proxy required to fit into other communication protocols such as BlueTooth or Zigbee

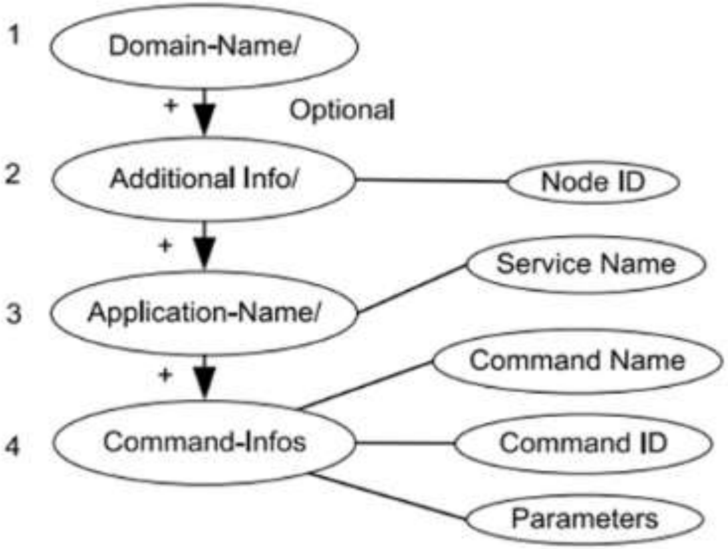


Naming : NDN (Named Data Networking)

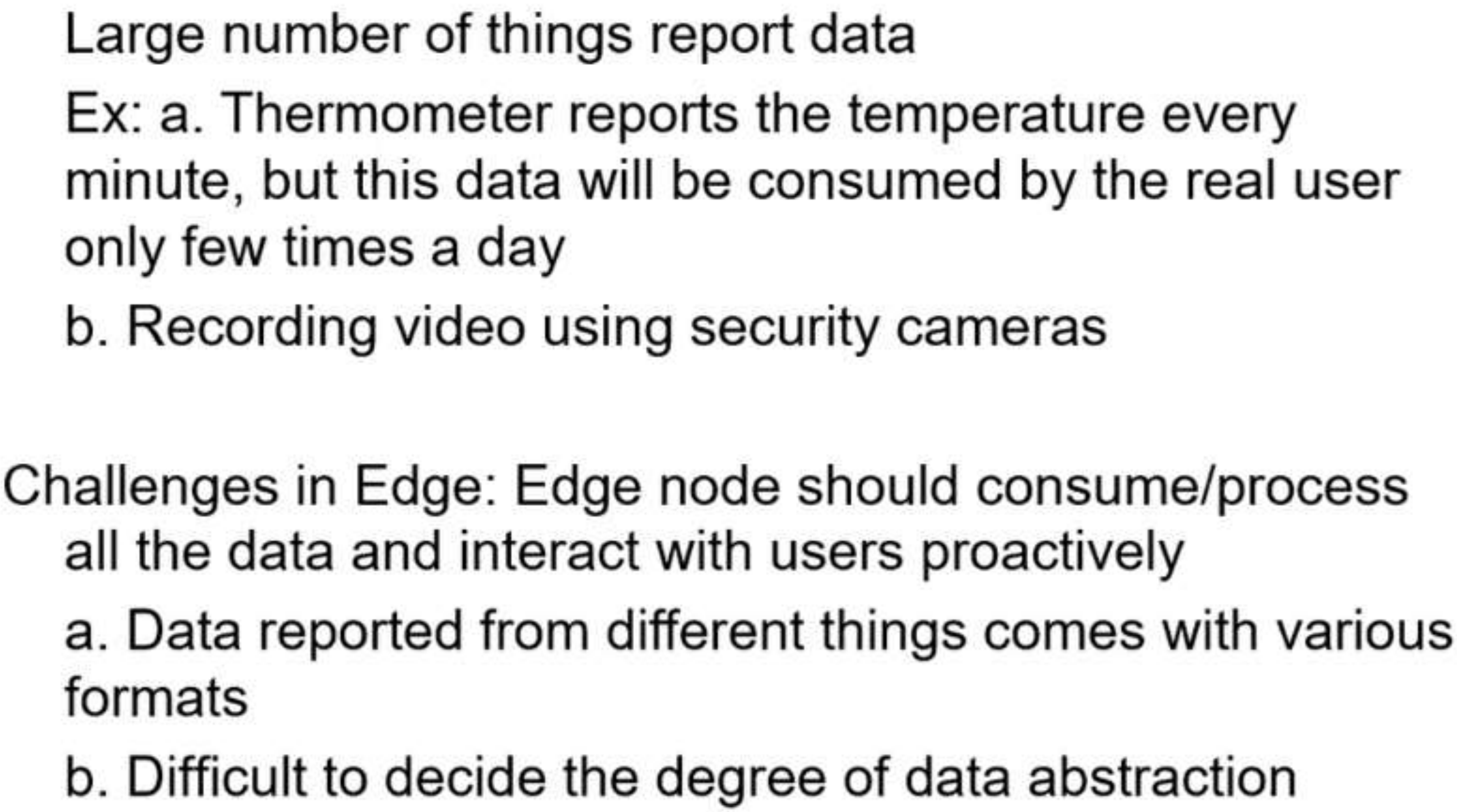
The data in IoT-NDN are addressed by names.

Requesting data is based on hierarchically structured approach.

The names contain human-friendly components and are location independent



Data Abstraction



Large number of things report data

Ex: a. Thermometer reports the temperature every minute, but this data will be consumed by the real user only few times a day

b. Recording video using security cameras

Challenges in Edge: Edge node should consume/process all the data and interact with users proactively

a. Data reported from different things comes with various formats

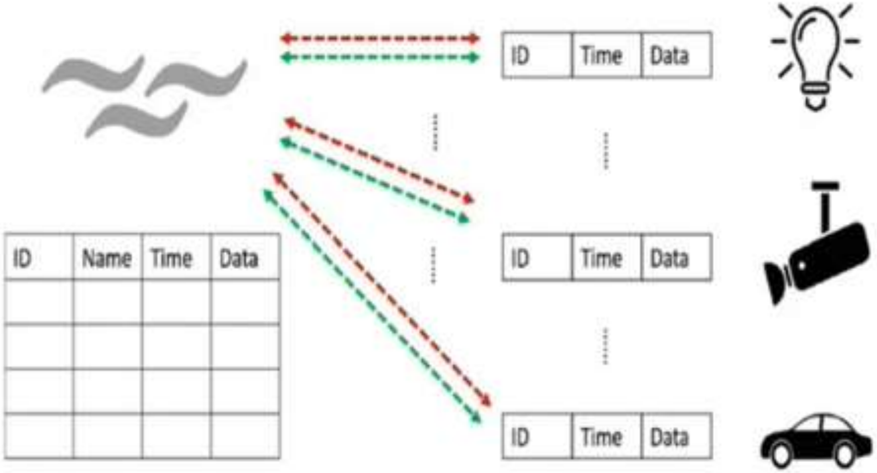
b. Difficult to decide the degree of data abstraction

Data Abstraction : Challenges in Edge

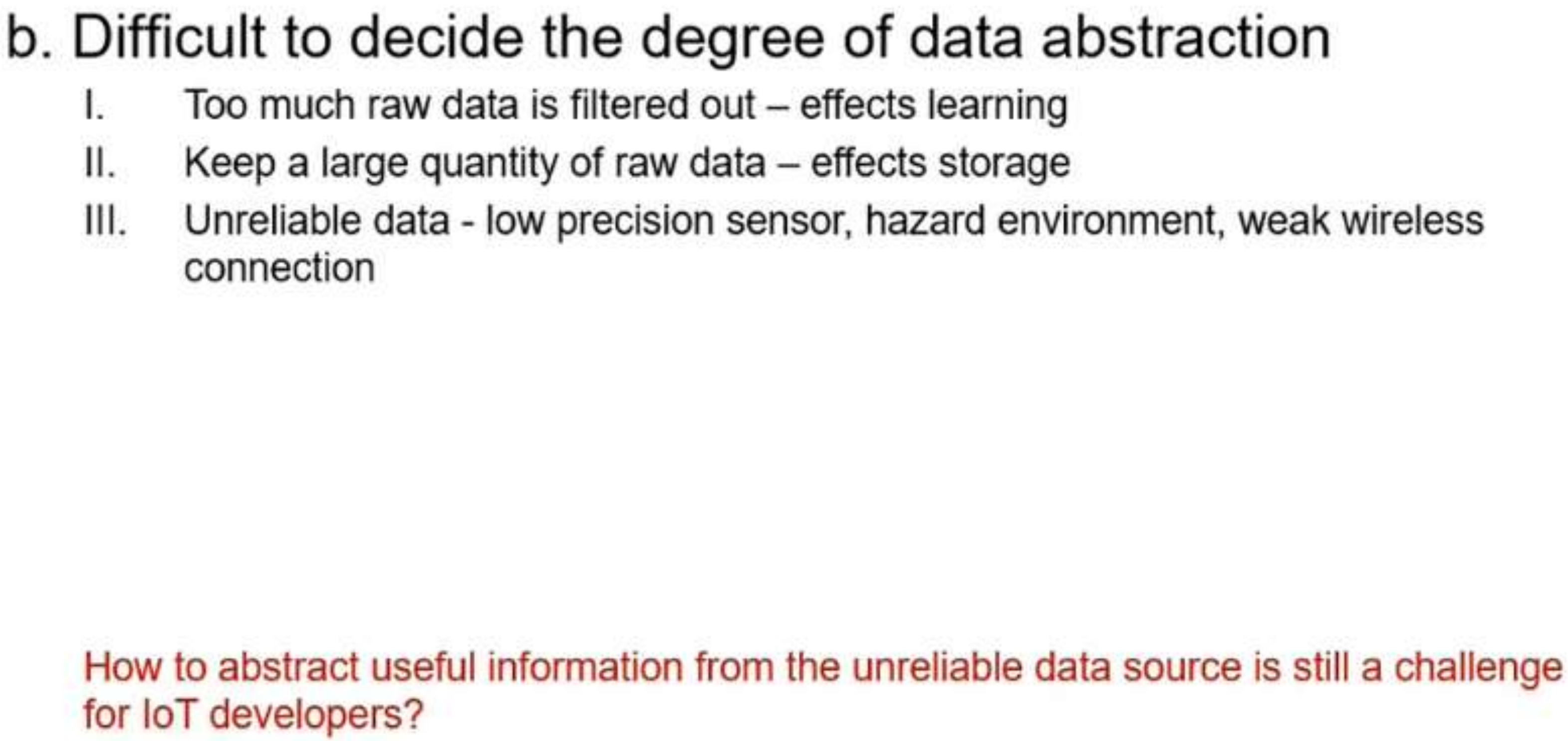
a. Data reported from different things comes with various formats

   I. Gateway should not see raw data

   II. Extract the knowledge from an integrated data table



Data Abstraction : Challenges in Edge



b. Difficult to decide the degree of data abstraction

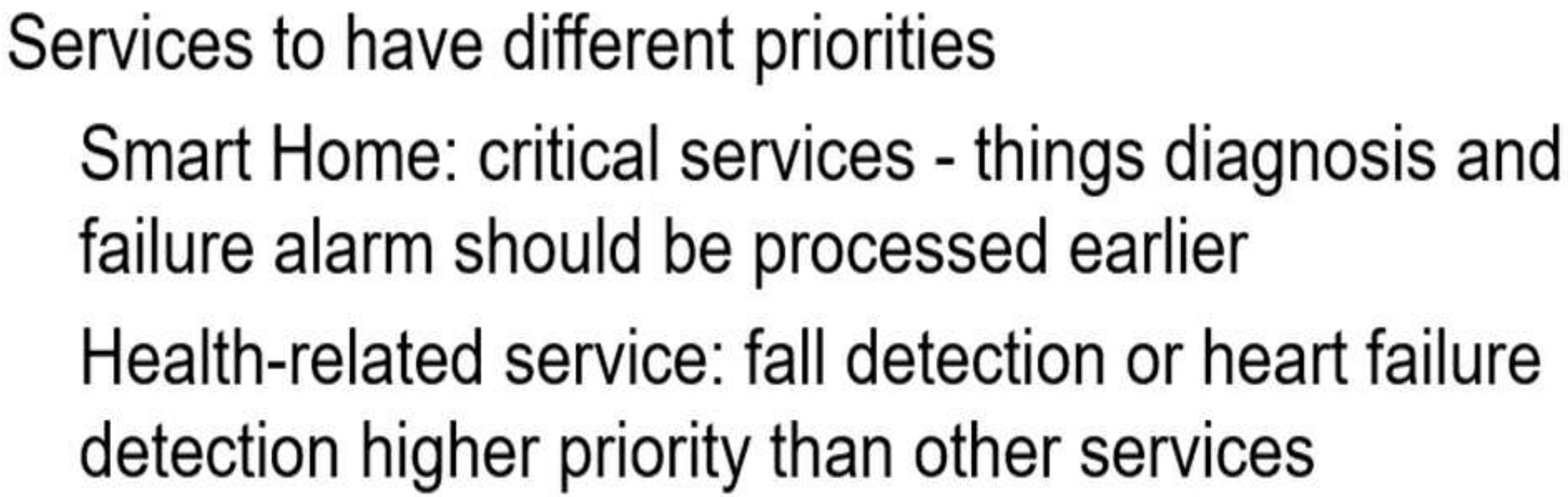
I. Too much raw data is filtered out – effects learning

II. Keep a large quantity of raw data – effects storage

III. Unreliable data - low precision sensor, hazard environment, weak wireless connection

How to abstract useful information from the unreliable data source is still a challenge for IoT developers?

Service Management: Differentiation



Services to have different priorities

- Smart Home: critical services - things diagnosis and failure alarm should be processed earlier
- Health-related service: fall detection or heart failure detection higher priority than other services

Service Management : Extensibility

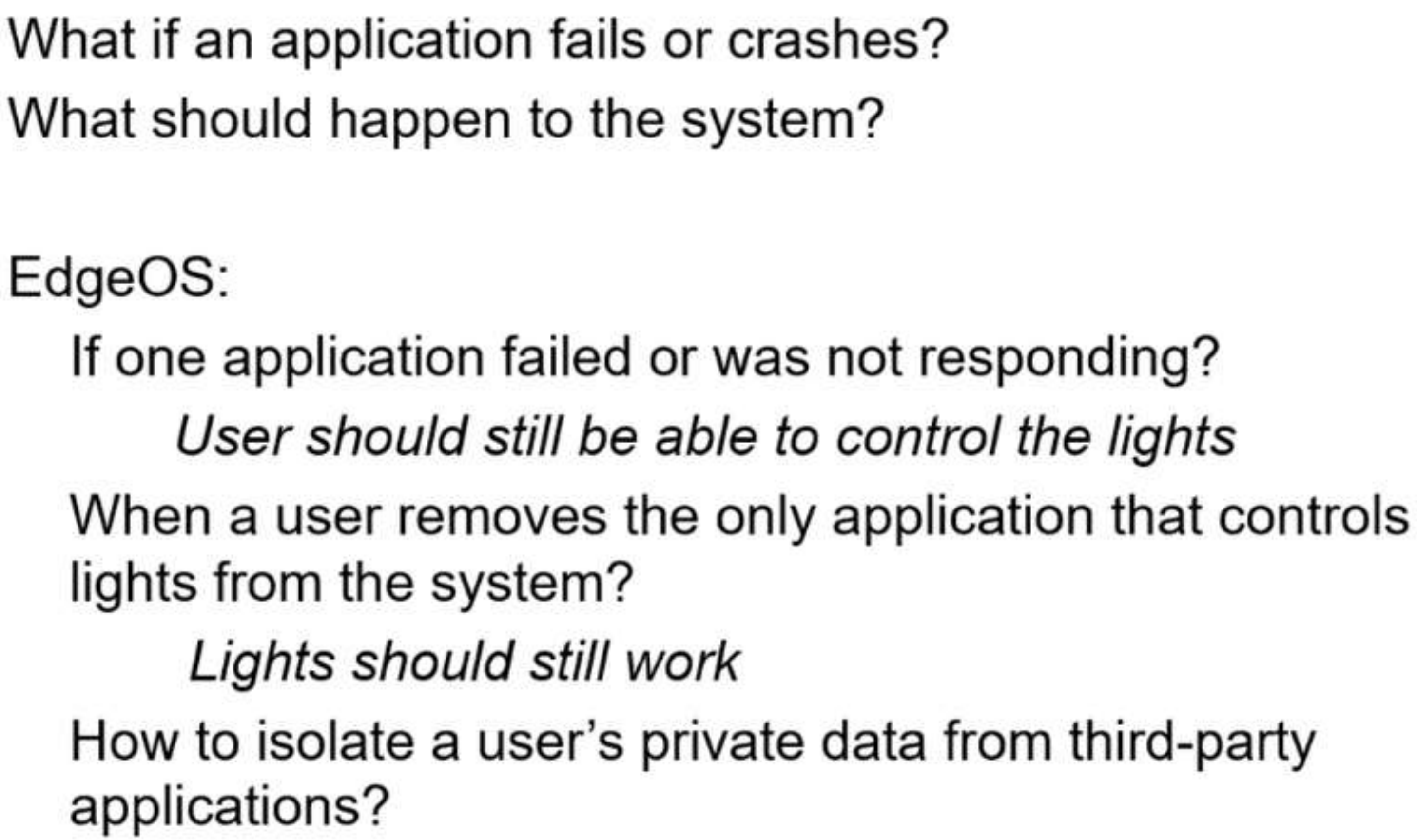
When you buy a new mobile device and connection?

Can a new ***Device*** be easily added to the current service without any problem?

Solution: to design service management layer

- flexible and extensible

Service Management : Isolation



What if an application fails or crashes?

What should happen to the system?

EdgeOS:

- If one application failed or was not responding?
  - *User should still be able to control the lights*
- When a user removes the only application that controls lights from the system?
  - *Lights should still work*
- How to isolate a user's private data from third-party applications?

Service Management : Reliability

From the service point of view

- Sometimes difficult to identify the reason for a service failure accurately
- Ex: if an air conditioner is not working
  - reasons:-
    - power cord is cut
    - compressor failure
    - temperature controller has run out of battery
    - sensor node lost connection to battery outage
    - bad connection condition
    - component wear out

Service Management : Reliability

- From the data point of view
  - Reliability in
    - data sensing
      - low battery
      - physical damage
    - data in communication
      - unreliable communication protocols
        - using HTTP for communication
        - using Message Queuing Telemetry transport (MQTT) - QoS0 (At most once)

Privacy and Security

User privacy and data security protection are the essential services

Private information can be learned from the sensed usage data

Computing at the edge of the data resource – decent method to protect privacy and data security

Challenges

- Awareness of privacy and security

WiFi networks security

49% of WiFi networks are unsecured

80% of routers set on default passwords

89% of public WiFi hotspots are unsecured

Devices like IP camera, health monitor, or even some WiFi enabled toys can be easily connected and misused

Challenges

- Ownership of the data collected from things @Edge

  Data collected by wearables - stored and analyzed at the service provider side

  Private photos and Videos

Storing data at the Edge device which is owned by the user provides better privacy protection

User should be able to control if service providers should use the data by process of authorization

Application Distribution

How to distribute the individual applications to various Edge nodes?

The current approaches for application distribution can be divided into two categories:

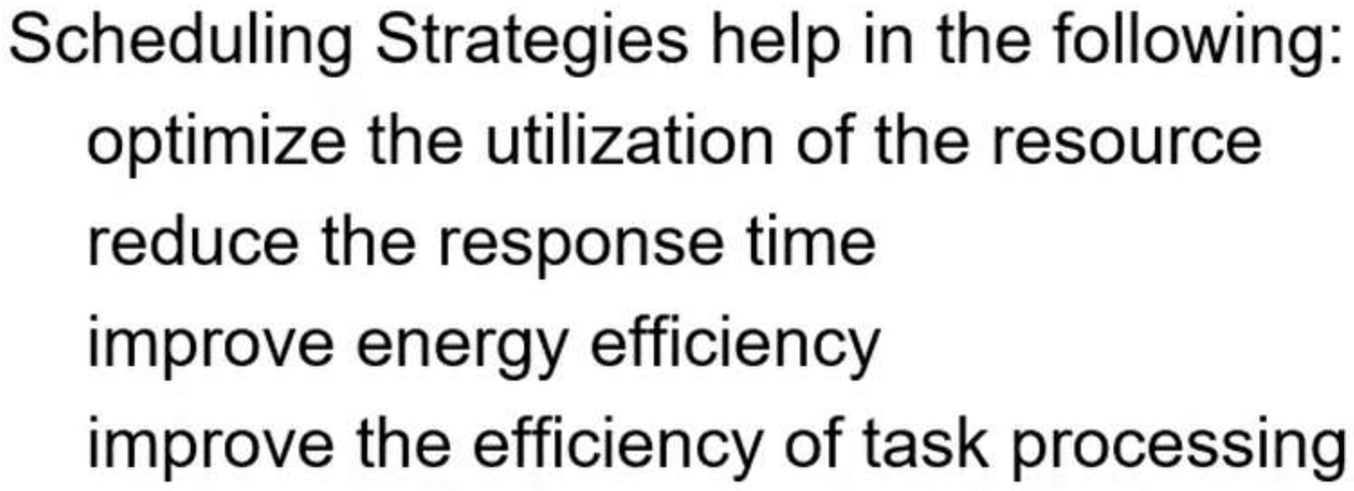
- dynamic: Hadoop distributed system
- static: Messing Passing Interface (MPI)

Application distribution approaches for Edge computing:

Cloud-Edge

Edge-Edge

Scheduling Strategies



Scheduling Strategies help in the following:

- optimize the utilization of the resource
- reduce the response time
- improve energy efficiency
- improve the efficiency of task processing

Scheduling Strategies

“One size does not fit all”

Scheduling strategies of Edge computing need to be designed according to:

- Different applications
- Based on the heterogeneous of the resources like:
  - Data
  - Computing
  - Storage
  - Network

Demo: EdgeCloudSim or PureEdgeSim

Business Model

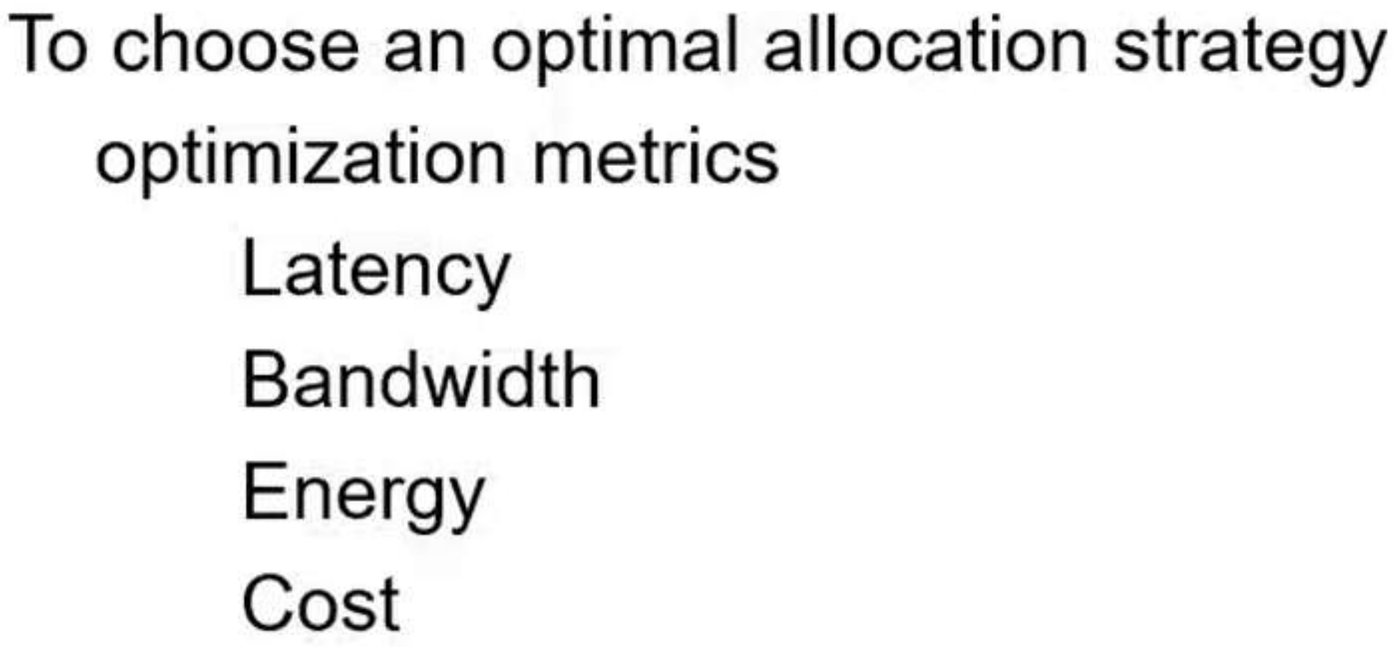
Business Model of Cloud computing – Simple, straight forward

- Users directly purchase service from the service provider

- Access it over Internet

- services could be IT infrastructure, software, and other resources

Optimization Metrics



- To choose an optimal allocation strategy
  - optimization metrics
    - Latency
    - Bandwidth
    - Energy
    - Cost