IIT KHARAGPUR

NPTEL ONLINE CERTIFICATION COURSES

NPTEL

Key Enablers of Industrial IoT: Connectivity – Part 4

Dr. Sudip Misra

Professor

Department of Computer Science and Engineering

Indian Institute of Technology Kharagpur

Email: smisra@sit.iitkgp.ernet.in

Website: http://cse.iitkgp.ac.in/~smisra/

Research Lab: cse.iitkgp.ac.in/~smisra/swan/

LPWAN

Introduction to LPWAN

- LPWAN stands for “Low Power Wide Area Network” is a wireless wide area network technology.
- Enables long range wireless communication among “things” at a low bit rate.
- It includes both standardized and proprietary solutions. Some of the technologies include LoRa, Sigfox's LPWAN.

LoRa and LoRaWAN

- LoRa, a short form for Long Range, incorporates a spread spectrum modulation technique based on chirp spread spectrum (CSS) technology.
- LoRa operates in the license-free sub-gigahertz radio frequency bands of 169 MHz, 433 MHz, 868 MHz (Europe) and 915 MHz (North America).
- LoRaWAN is the network in which LoRa operates and enables communication between devices.

Source: What is LoRa?.

SIGFOX

- The SIGFOX network and technology achieves low cost wide coverage for application domains with machine to machine networking and communication.
- The SIGFOX radio link operates in the unlicensed ISM radio bands.
- SIGFOX network give a performance of upto 140 messages per day with a payload of 12 bytes per message.
- The wireless throughput achieved is of up to 100 bits per second.

Source: *Ian Poole.* SIGFOX for M2M & IoT

Hands-On
(Industrial Environment Monitoring)

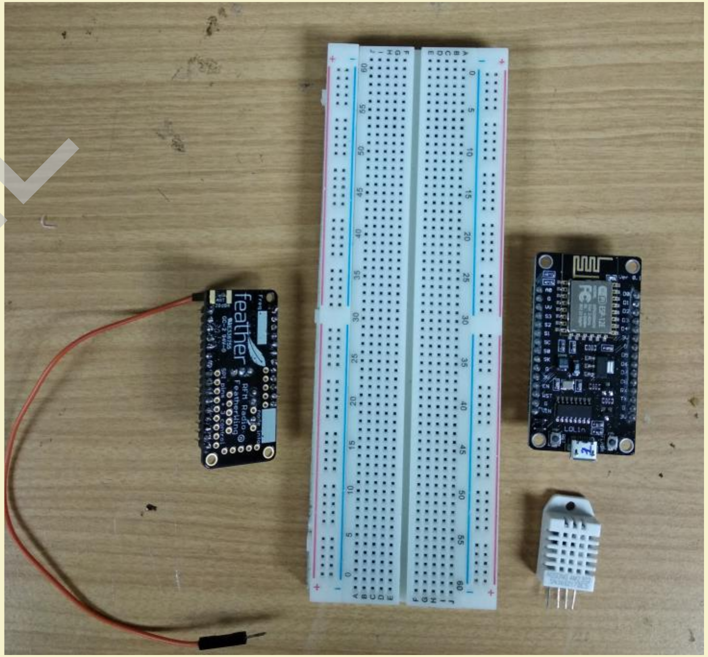
System Overview

- Sensor (DHT) and Communication Module (LoRa) interfaced with Processor (NodeMCU)
- Both transmitter and receiver module consists of a NodeMCU board connected to a LoRa module.
- Transmitter module has the sensor that monitors the temperature and humidity of the environment and sends the data to the receiver module.
- Receiver module responds according to the set condition.

System Overview (contd.)

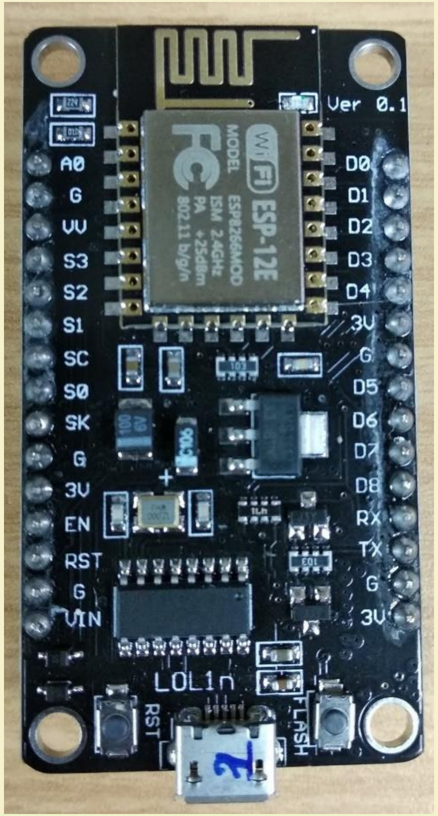
Requirements:

- NodeMCU
- LoRa
- DHT Sensor
- Jumper wires
- LED



NodeMCU

- This is an ESP-12 module and works with Arduino IDE.
- We can use other Arduino Boards as well.
- Pin configuration along with other documentation can be found here.

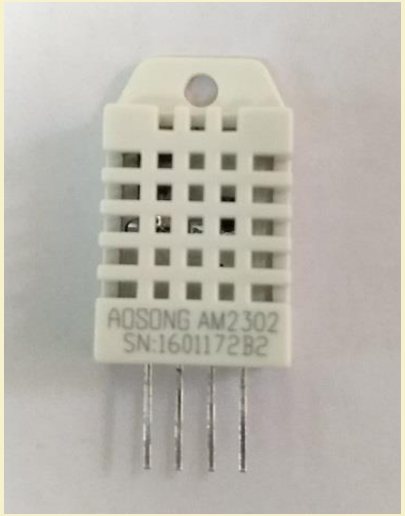


LoRa

- This is a LoRa transceiver module as discussed in the previous slides.
- It is used for long range wireless communication in industrial applications.

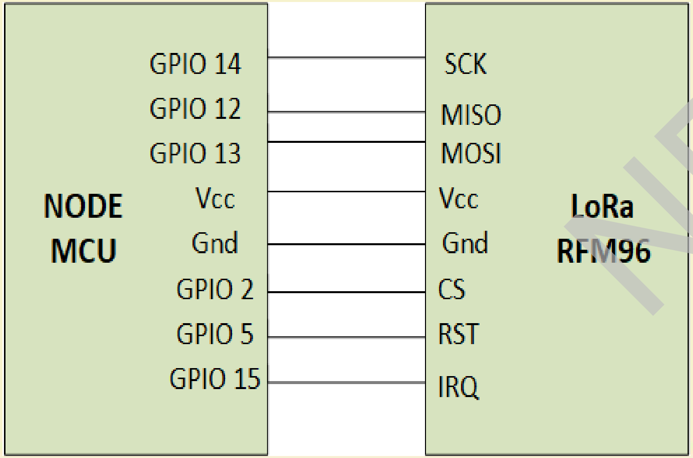
DHT Sensor

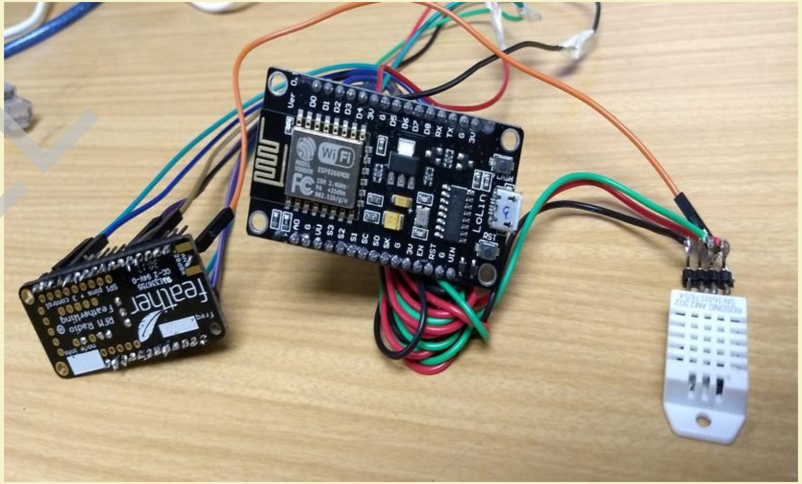
- Digital Humidity and Temperature (DHT) Sensor
- Pin Configuration (from left to right)
  - PIN 1- 3.3V-5V Power supply
  - PIN 2- Data
  - PIN 3- Null
  - PIN 4- Ground



Interfacing

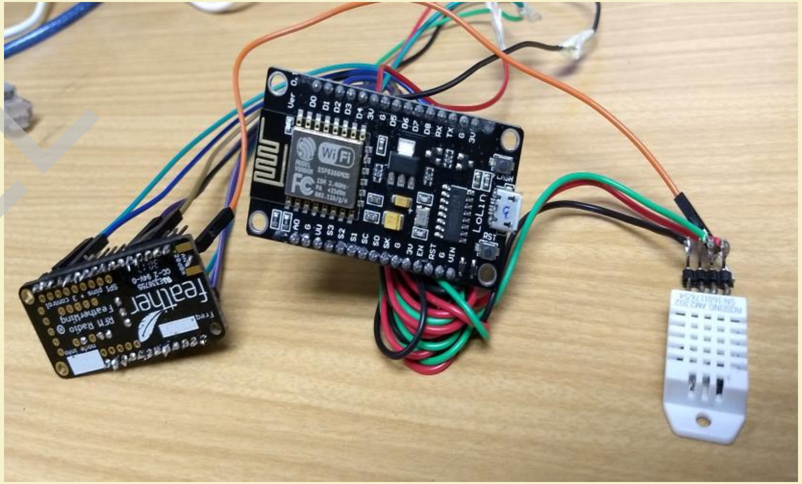
- The connection between NodeMCU and LoRa is shown in the diagram.





Interfacing

- The connection between NodeMCU and DHT is shown in the diagram.
- NodeMCU ---- DHT
  - GPIO 4 – Data
  - 3V3 – Vcc
  - Gnd – Gnd



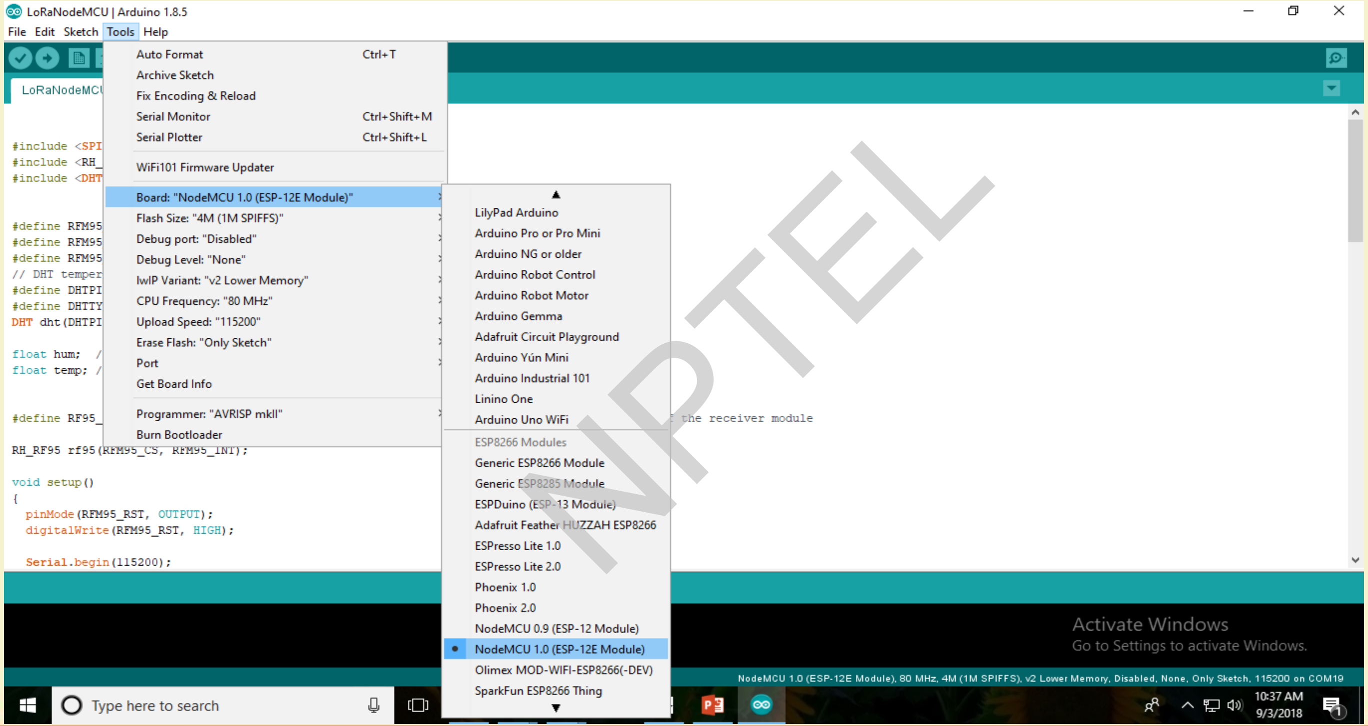
Pre-requisites

- Adafruit provides a library to work with the DHT22 sensor.
- To work with LoRa we use the Radiohead library which can be downloaded from the below URL.
  - https://learn.adafruit.com/radio-featherwing/using-the-rfm-9x-radio
- The initial connections have to be soldered in the LoRa module as mentioned in the URL provided above.

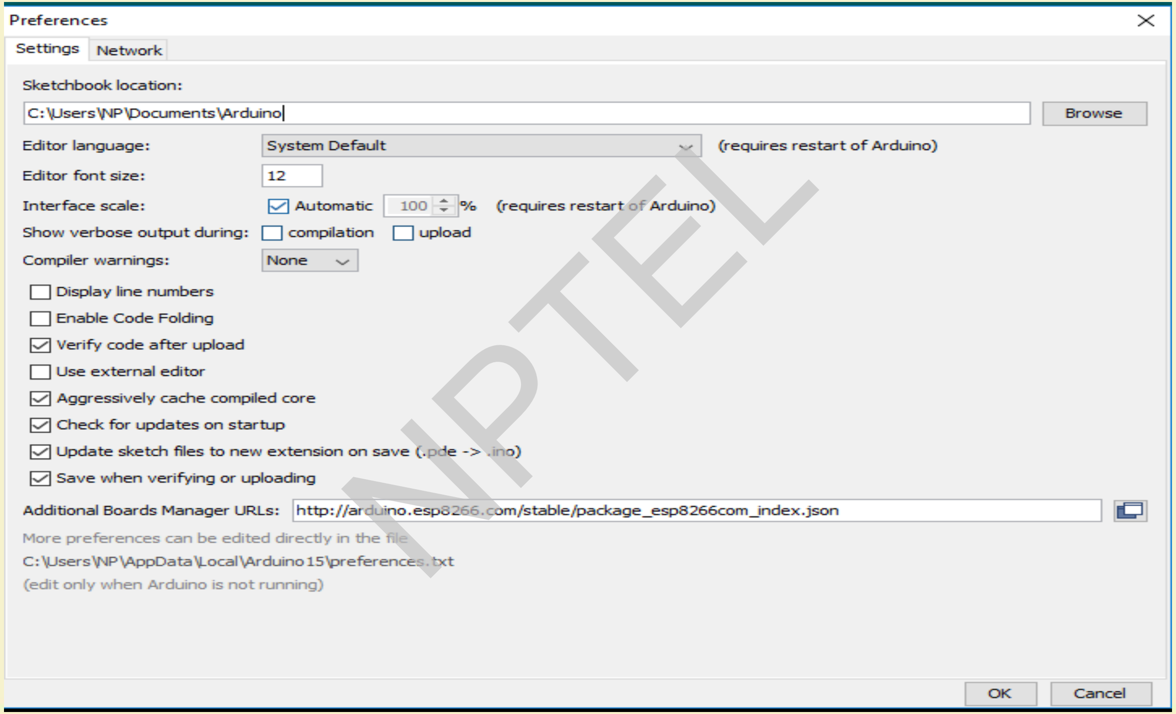
Pre-requisites (contd.)

- To add Node MCU board in the Arduino IDE, follow the below steps:
  - Arduino IDE >> File >> Preferences (Shortcut is CTRL + COMMA)>> Settings tab >> on Additional Board Manager URL side type this >>
  - http://arduino.esp8266.com/stable/package_esp8266com_index.json
  - click ok

Pre-requisites (contd.)



Pre-requisites (contd.)



Program: LoRa interfaced with NodeMCU

LoRaNodeMCUTx

```
#include <SPI.h>
#include <RH_RF95.h>
#include <DHT.h>

#define CS  2    // "E" D4
#define RST 5   // "D" D1
#define INT 15   // "B" D8
// DHT temperature and humidity sensor
#define DHTPIN 4     // Pin numbers in GPIO/D2
#define DHTTYPE DHT22   // DHT 22
DHT dht(DHTPIN, DHTTYPE);

float hum;  //Stores humidity value
float temp; //Stores temperature value


#define FREQ 915.0 // This can be changed to other frequency but should be same as that of the receiver module

RH_RF95 rf95(CS, INT);
```

- Here we declare the pins for connection with the CS, RST and IRQ pin of LoRa.
- The DHT data pin is mapped with the Node MCU pin.

Program: LoRa interfaced with NodeMCU(Tx)

```
//Reading data from the DHT sensor
hum = dht.readHumidity();
temp= dht.readTemperature();
String msg1= "Temp: ";
msg1 += temp;
msg1 += "C, Hum: ";
msg1 += hum;
msg1 += "%";
delay(1000); // Delay of 1 second before transmitting the data
Serial.println("Sending temperature and humidity");

//Send data to the receiver
char radiopacket[26];
msg1.toCharArray(radiopacket,26);
Serial.println(radiopacket);
delay(10);
rf95.send((uint8_t *)radiopacket, 26);
```

- The temperature and humidity value from the sensor is read and saved in a string.
- The data is sent to the receiver module in a character array with a delay of 1 second.

Program: LoRa interfaced with NodeMCU(Rx)

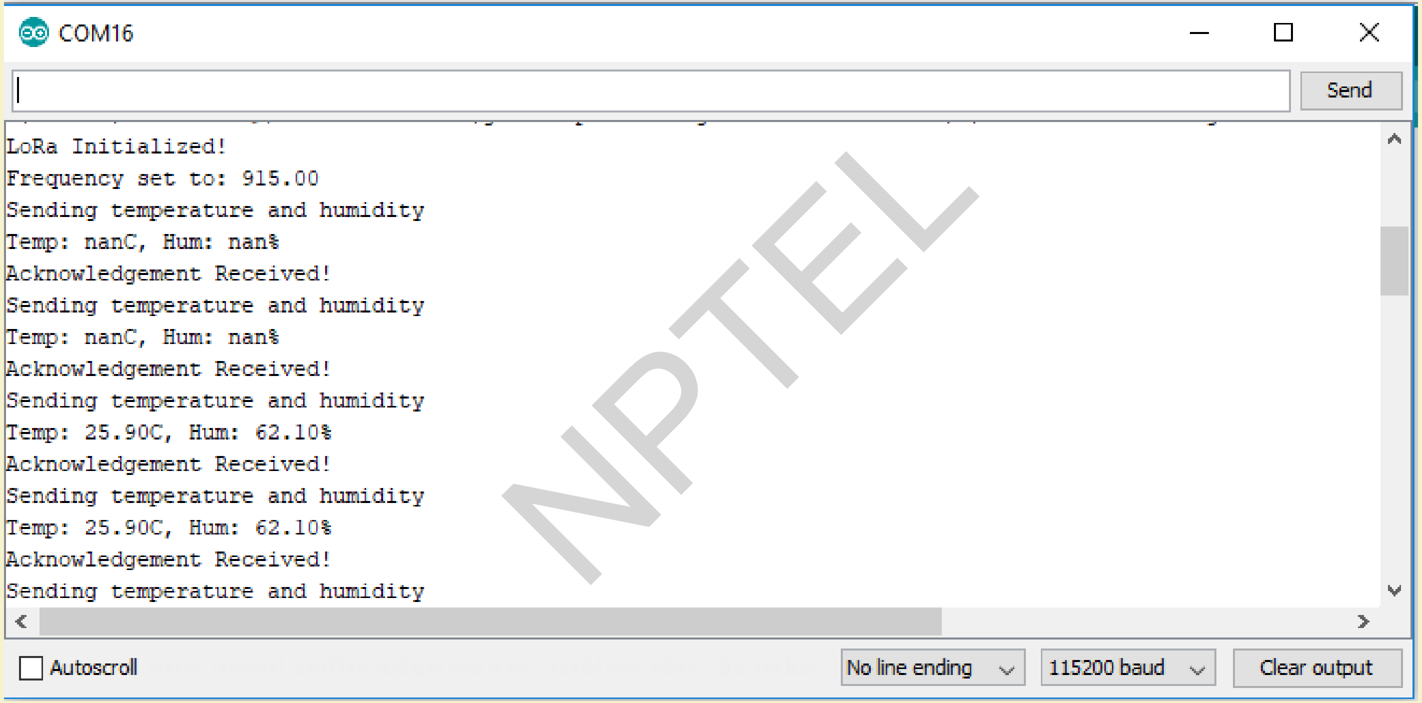
LoRaNodeMCURx

```
#include <SPI.h>
#include <RH_RF95.h>

#define CS  2    // "E"
#define RST 5   // "D"
#define INT 15   // "B"

#define FREQ 915.0

RH_RF95 rf95(CS, INT);

#define LED 4 //GPIO4- D2
```

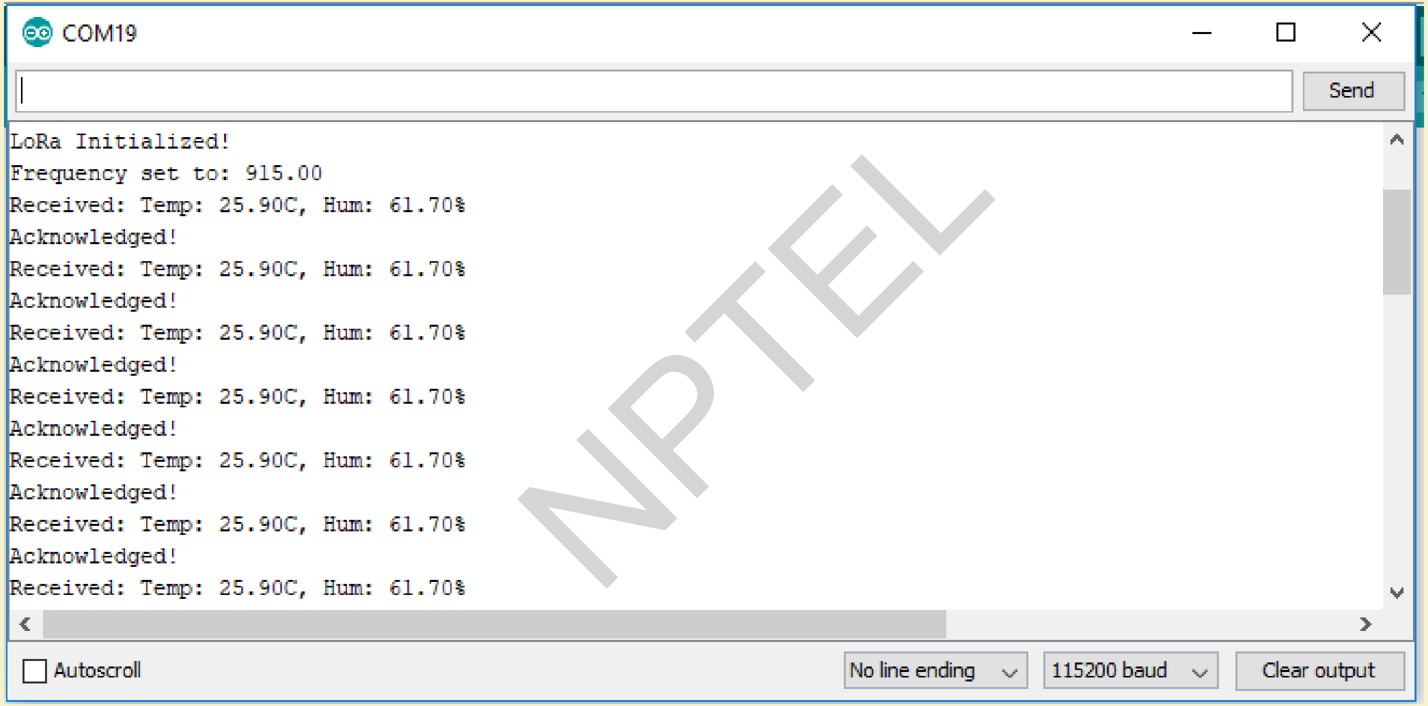
```
void loop()
{
  if (rf95.available())
  {
    uint8_t buf[RH_RF95_MAX_MESSAGE_LEN];
    uint8_t len = sizeof(buf);

    if (rf95.recv(buf, &len))
    {
      digitalWrite(LED, HIGH);
      Serial.print("Received: ");
      Serial.println((char*)buf);

      // Send a reply
      uint8_t data[] = "Data Received";
      rf95.send(data, sizeof(data));
      rf95.waitPacketSent();
      Serial.println("Acknowledged!");
      digitalWrite(LED, LOW);
```

- The data is received by the Receiver module.
- After successful reception, an acknowledgement message is sent to the sender module.
- Every time a message is received, the LED pin is set to HIGH.

Output from Tx Serial Monitor



Output from Rx Serial Monitor



References

1. Industrial Internet of Things: IIoT communication and connectivity technology 2017. Online. URL: https://www.i-scoop.eu/internet-of-things-guide/industrial-internet-things-iiot-saving-costs-innovation/iiot-connectivity-connections/
2. What is LoRa?. Online. URL: https://www.semtech.com/lora/what-is-lora

Tx Program: LoRa interfaced with NodeMCU

```
#include <SPI.h>
#include <RH_RF95.h>
#include <DHT.h>

#define CS  2    // "E" D4
#define RST 5   // "D" D1
#define INT 15   // "B" D8
// DHT temperature and humidity sensor
#define DHTPIN 4      // Pin numbers in
GPIO/D2
#define DHTTYPE DHT22   // DHT 22
DHT dht(DHTPIN, DHTTYPE);

float hum;  //Stores humidity value
float temp; //Stores temperature value

#define FREQ 915.0
//Can be changed to other freq but should be
same as that of the Rx

RH_RF95 rf95(CS, INT);

void setup()
{
  pinMode(RST, OUTPUT);
  digitalWrite(RST, HIGH);

  Serial.begin(115200);
  while (!Serial) {
    delay(1);
  }
  delay(100);
  Serial.println("LoRa Tx Node");

  // manual reset
  digitalWrite(RST, LOW);
  delay(10);
  digitalWrite(RST, HIGH);
  delay(10);

  while (!rf95.init()) {
    Serial.println("Initialization Failed!");
    while (1);
  }
  Serial.println("LoRa Initialized!");

  if (!rf95.setFrequency(FREQ)) {
    Serial.println("setFrequency failed");
    while (1);
  }
  Serial.print("Frequency set to: ");
Serial.println(FREQ);
  rf95.setTxPower(23, false);
}
```

Tx Program: LoRa interfaced with NodeMCU

```
void loop()
{
  //Reading data from the DHT sensor
  hum = dht.readHumidity();
  temp= dht.readTemperature();
  String msg1= "Temp: ";
  msg1 += temp;
  msg1 += "C, Hum: ";
  msg1 += hum;
  msg1 += "%";
  delay(1000); // Delay of 1 second before
transmitting the data
  Serial.println("Sending temperature and
humidity");

 //Send data to the receiver
  char radiopacket[26];
  msg1.toCharArray(radiopacket,26);
  Serial.println(radiopacket);
  delay(10);
  rf95.send((uint8_t *)radiopacket, 26);
  delay(10);
  rf95.waitPacketSent();
  uint8_t buf[RH_RF95_MAX_MESSAGE_LEN];
  uint8_t len = sizeof(buf);

 if (rf95.waitAvailableTimeout(1000))
  {
   if (rf95.recv(buf, &len))
   {
     Serial.print("Acknowledgement
Received!\n");
    }
   else
   {
     Serial.println("Receive failed\n");
    }
  }
  else
  {
   Serial.println("No Receiver Node Found!");
  }
}
```

Rx Program: LoRa interfaced with NodeMCU

```
#include <SPI.h>
#include <RH_RF95.h>

#define CS  2    // "E"
#define RST 5   // "D"
#define INT 15   // "B"

#define FREQ 915.0

RH_RF95 rf95(CS, INT);

#define LED 4 //GPIO4- D2

void setup()
{
  pinMode(LED, OUTPUT);
  pinMode(RST, OUTPUT);
  digitalWrite(RST, HIGH);
Serial.begin(115200);
while (!Serial) {
  delay(1);
}
delay(100);

Serial.println("LoRa Rx Node");
digitalWrite(RST, LOW);//Reset manually
delay(10);
digitalWrite(RST, HIGH);
delay(10);

while (!rf95.init()) {
  Serial.println("Initialization Failed!");
  while (1);
}
Serial.println("LoRa Initialized!");

if (!rf95.setFrequency(FREQ)) {
    Serial.println("setFrequency failed");
    while (1);
  }
  Serial.print("Frequency set to: ");
Serial.println(FREQ);

  rf95.setTxPower(23, false);
}
```

Rx Program: LoRa interfaced with NodeMCU

```
void loop()
{
  if (rf95.available())
  {
    uint8_t
buf[RH_RF95_MAX_MESSAGE_LEN];
    uint8_t len = sizeof(buf);

    if (rf95.recv(buf, &len))
    {
      digitalWrite(LED, HIGH);
      //RH_RF95::printBuffer("Received: ", buf,
len);
      Serial.print("Received: ");
      Serial.println((char*)buf);

      // Send a reply
      uint8_t data[] = "Data Received";
      rf95.send(data, sizeof(data));
      rf95.waitPacketSent();
      Serial.println("Acknowledged!");
      digitalWrite(LED, LOW);
    }
    else
    {
      Serial.println("Receive failed");
    }
  }
}
```

Thank You!!

Key Enablers of Industrial IoT: Connectivity – Part 4

Dr. Sudip Misra
Professor
Department of Computer Science and Engineering
Indian Institute of Technology Kharagpur
Email: smisra@sit.iitkgp.ernet.in
Website: http://cse.iitkgp.ac.in/~smisra/
Research Lab: cse.iitkgp.ac.in/~smisra/swan/

LPWAN

Introduction to LPWAN

- LPWAN stands for “Low Power Wide Area Network” is a wireless wide area network technology.
- Enables long range wireless communication among “things” at a low bit rate.
- It includes both standardized and proprietary solutions. Some of the technologies include LoRa, Sigfox's LPWAN.

LoRa and LoRaWAN

- LoRa, a short form for Long Range, incorporates a spread spectrum modulation technique based on chirp spread spectrum (CSS) technology.
- LoRa operates in the license-free sub-gigahertz radio frequency bands of 169 MHz, 433 MHz, 868 MHz (Europe) and 915 MHz (North America).
- LoRaWAN is the network in which LoRa operates and enables communication between devices.

Source: What is LoRa?.

SIGFOX

- The SIGFOX network and technology achieves low cost wide coverage for application domains with machine to machine networking and communication.
- The SIGFOX radio link operates in the unlicensed ISM radio bands.
- SIGFOX network give a performance of upto 140 messages per day with a payload of 12 bytes per message.
- The wireless throughput achieved is of up to 100 bits per second.

Source: *Ian Poole.* SIGFOX for M2M & IoT

Hands-On
(Industrial Environment Monitoring)

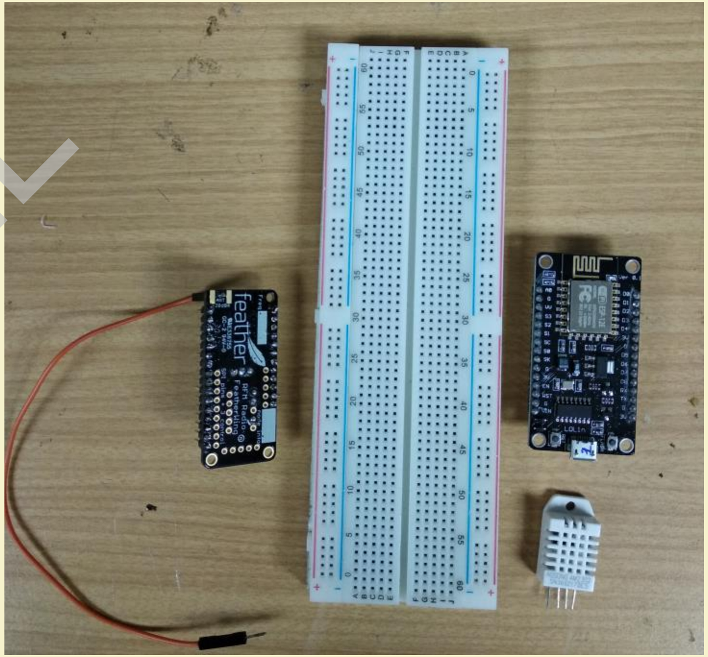
System Overview

- Sensor (DHT) and Communication Module (LoRa) interfaced with Processor (NodeMCU)
- Both transmitter and receiver module consists of a NodeMCU board connected to a LoRa module.
- Transmitter module has the sensor that monitors the temperature and humidity of the environment and sends the data to the receiver module.
- Receiver module responds according to the set condition.

System Overview (contd.)

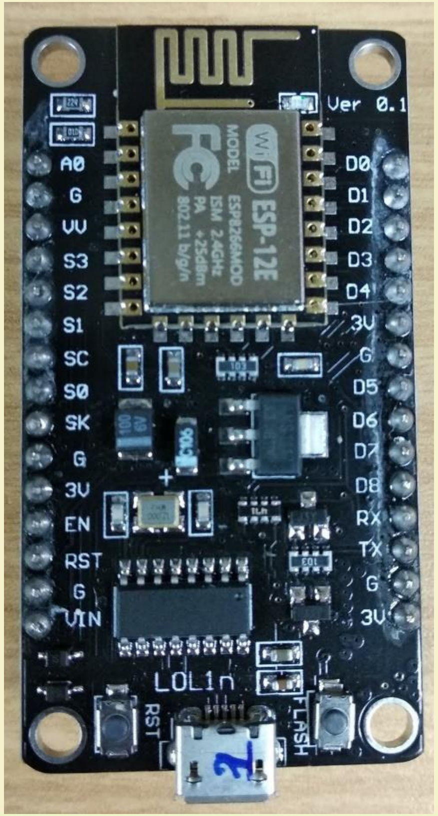
Requirements:

- NodeMCU
- LoRa
- DHT Sensor
- Jumper wires
- LED



NodeMCU

- This is an ESP-12 module and works with Arduino IDE.
- We can use other Arduino Boards as well.
- Pin configuration along with other documentation can be found here.

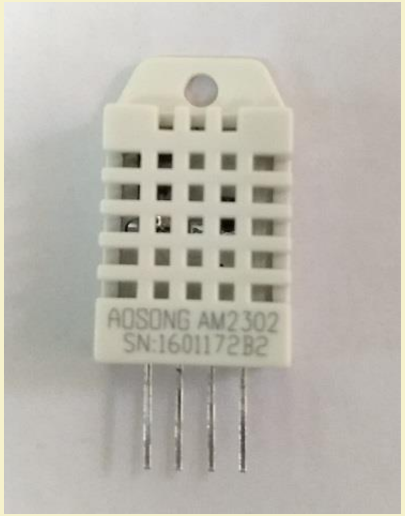


LoRa

- This is a LoRa transceiver module as discussed in the previous slides.
- It is used for long range wireless communication in industrial applications.

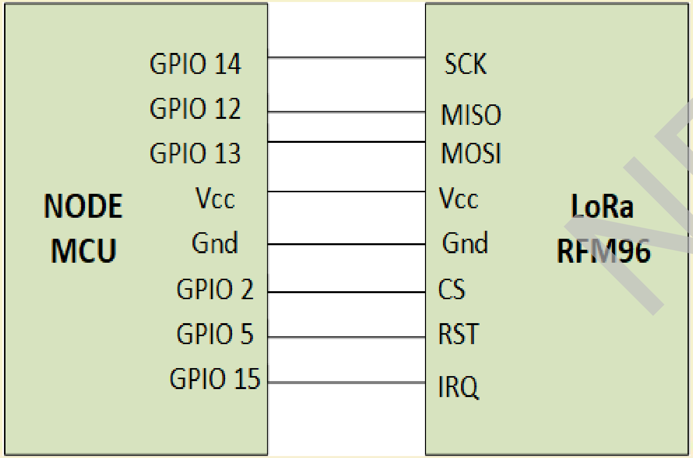
DHT Sensor

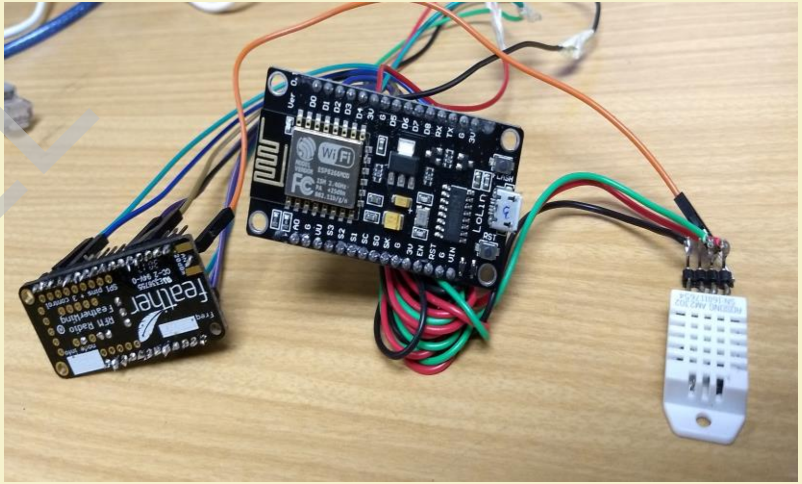
- Digital Humidity and Temperature (DHT) Sensor
- Pin Configuration (from left to right)
  - PIN 1- 3.3V-5V Power supply
  - PIN 2- Data
  - PIN 3- Null
  - PIN 4- Ground



Interfacing

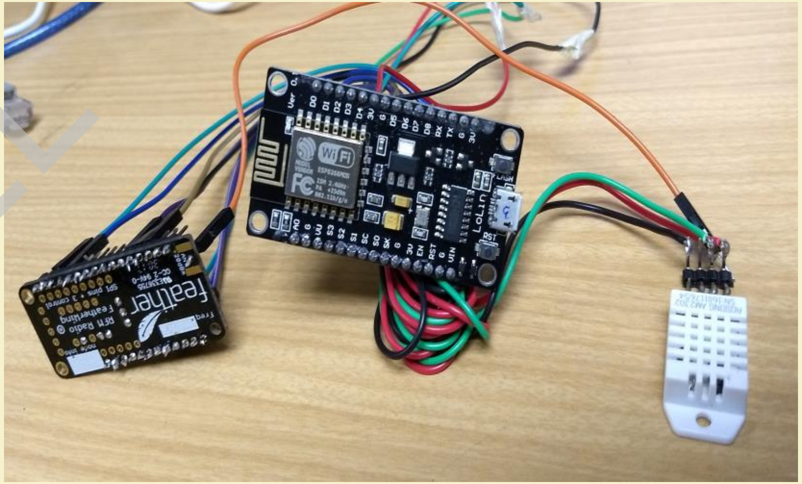
- The connection between NodeMCU and LoRa is shown in the diagram.





Interfacing

- The connection between NodeMCU and DHT is shown in the diagram.
- NodeMCU ---- DHT
  - GPIO 4 – Data
  - 3V3 – Vcc
  - Gnd – Gnd



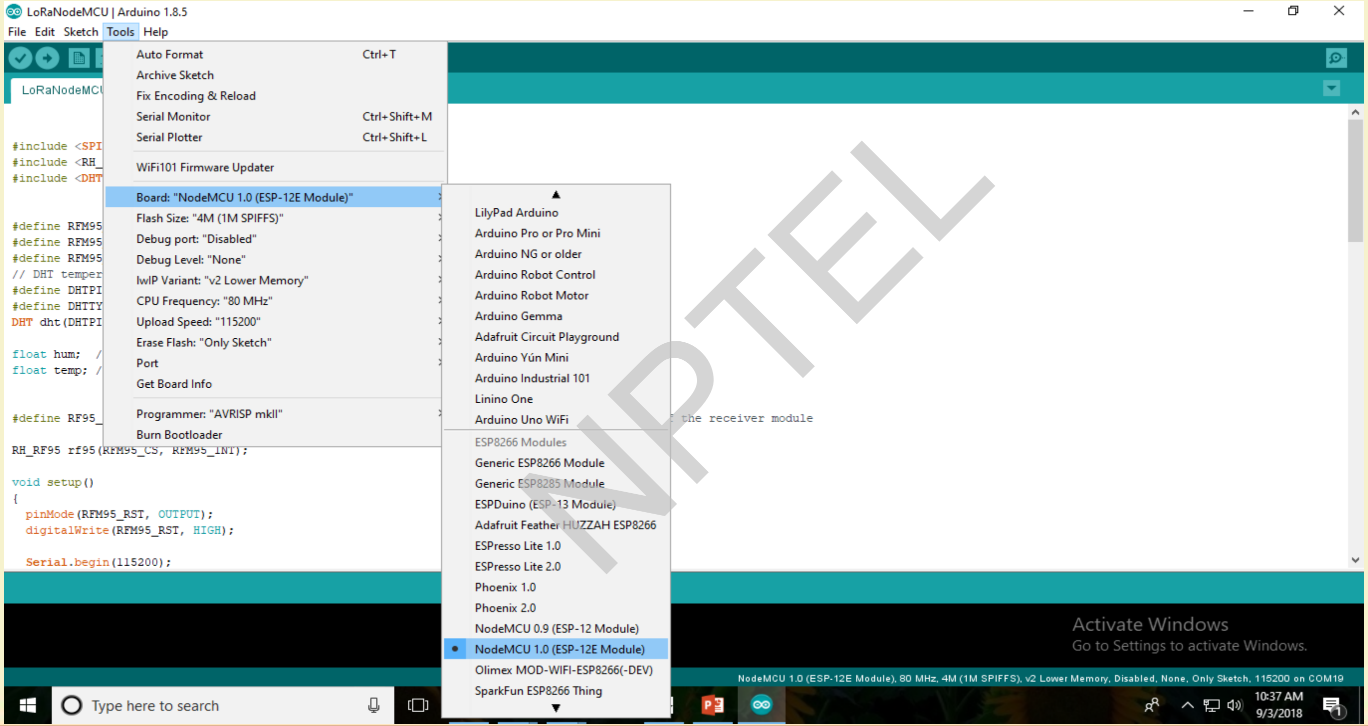
Pre-requisites

- Adafruit provides a library to work with the DHT22 sensor.
- To work with LoRa we use the Radiohead library which can be downloaded from the below URL.
  - https://learn.adafruit.com/radio-featherwing/using-the-rfm-9x-radio
- The initial connections have to be soldered in the LoRa module as mentioned in the URL provided above.

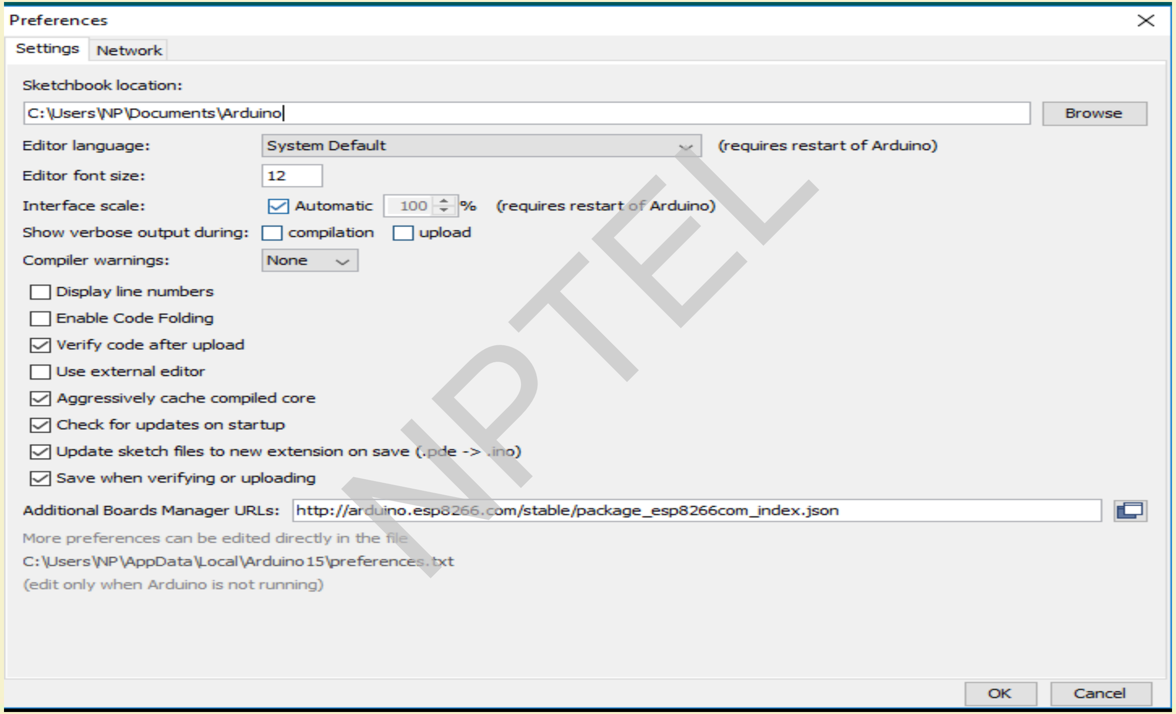
Pre-requisites (contd.)

- To add Node MCU board in the Arduino IDE, follow the below steps:
  - Arduino IDE >> File >> Preferences (Shortcut is CTRL + COMMA)>> Settings tab >> on Additional Board Manager URL side type this >>
  - http://arduino.esp8266.com/stable/package_esp8266com_index.json
  - click ok

Pre-requisites (contd.)



Pre-requisites (contd.)



Program: LoRa interfaced with NodeMCU

LoRaNodeMCUTx

```
#include <SPI.h>
#include <RH_RF95.h>
#include <DHT.h>

#define CS  2    // "E" D4
#define RST 5   // "D" D1
#define INT 15   // "B" D8
// DHT temperature and humidity sensor
#define DHTPIN 4     // Pin numbers in GPIO/D2
#define DHTTYPE DHT22   // DHT 22
DHT dht(DHTPIN, DHTTYPE);

float hum;  //Stores humidity value
float temp; //Stores temperature value


#define FREQ 915.0 // This can be changed to other frequency but should be same as that of the receiver module

RH_RF95 rf95(CS, INT);
```

- Here we declare the pins for connection with the CS, RST and IRQ pin of LoRa.
- The DHT data pin is mapped with the Node MCU pin.

Program: LoRa interfaced with NodeMCU(Tx)

```
//Reading data from the DHT sensor
hum = dht.readHumidity();
temp= dht.readTemperature();
String msg1= "Temp: ";
msg1 += temp;
msg1 += "C, Hum: ";
msg1 += hum;
msg1 += "%";
delay(1000); // Delay of 1 second before transmitting the data
Serial.println("Sending temperature and humidity");

//Send data to the receiver
char radiopacket[26];
msg1.toCharArray(radiopacket,26);
Serial.println(radiopacket);
delay(10);
rf95.send((uint8_t *)radiopacket, 26);
```

- The temperature and humidity value from the sensor is read and saved in a string.
- The data is sent to the receiver module in a character array with a delay of 1 second.

Program: LoRa interfaced with NodeMCU(Rx)

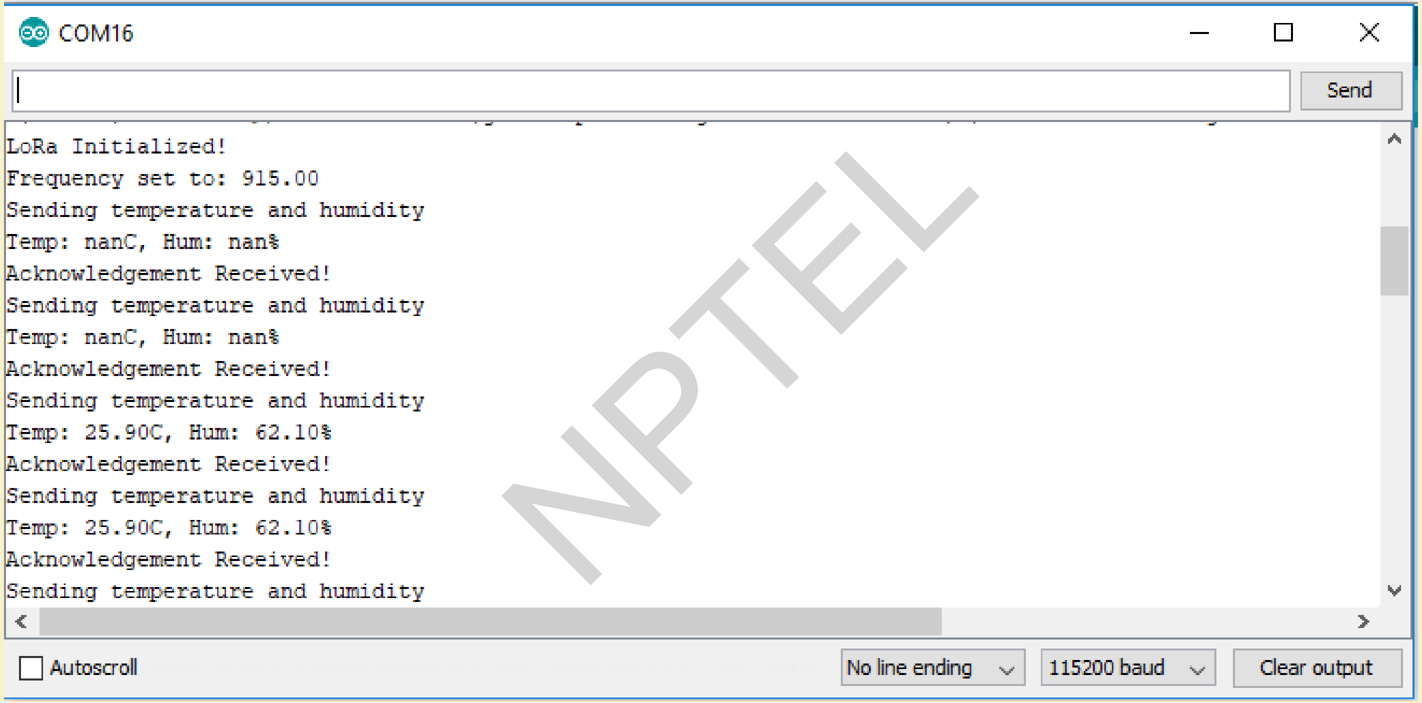
LoRaNodeMCURx

```
#include <SPI.h>
#include <RH_RF95.h>

#define CS  2    // "E"
#define RST 5   // "D"
#define INT 15   // "B"

#define FREQ 915.0

RH_RF95 rf95(CS, INT);

#define LED 4 //GPIO4- D2
```

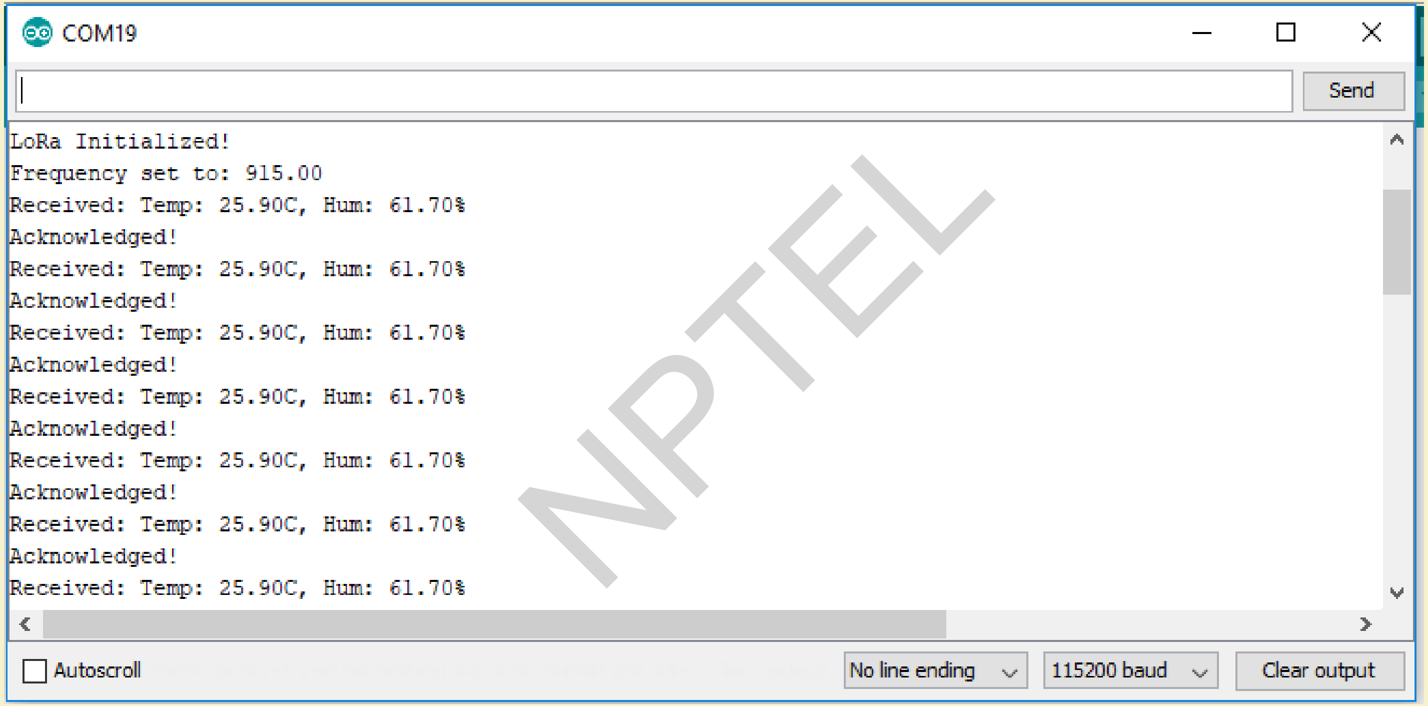
```
void loop()
{
  if (rf95.available())
  {
    uint8_t buf[RH_RF95_MAX_MESSAGE_LEN];
    uint8_t len = sizeof(buf);

    if (rf95.recv(buf, &len))
    {
      digitalWrite(LED, HIGH);
      Serial.print("Received: ");
      Serial.println((char*)buf);

      // Send a reply
      uint8_t data[] = "Data Received";
      rf95.send(data, sizeof(data));
      rf95.waitPacketSent();
      Serial.println("Acknowledged!");
      digitalWrite(LED, LOW);
```

- The data is received by the Receiver module.
- After successful reception, an acknowledgement message is sent to the sender module.
- Every time a message is received, the LED pin is set to HIGH.

Output from Tx Serial Monitor



Output from Rx Serial Monitor



References

1. Industrial Internet of Things: IIoT communication and connectivity technology 2017. Online. URL: https://www.i-scoop.eu/internet-of-things-guide/industrial-internet-things-iiot-saving-costs-innovation/iiot-connectivity-connections/
2. What is LoRa?. Online. URL: https://www.semtech.com/lora/what-is-lora

Tx Program: LoRa interfaced with NodeMCU

```
#include <SPI.h>
#include <RH_RF95.h>
#include <DHT.h>

#define CS  2    // "E" D4
#define RST 5   // "D" D1
#define INT 15   // "B" D8
// DHT temperature and humidity sensor
#define DHTPIN 4     // Pin numbers in
GPIO/D2
#define DHTTYPE DHT22   // DHT 22
DHT dht(DHTPIN, DHTTYPE);

float hum;  //Stores humidity value
float temp; //Stores temperature value

#define FREQ 915.0
//Can be changed to other freq but should be
same as that of the Rx

RH_RF95 rf95(CS, INT);

void setup()
{
  pinMode(RST, OUTPUT);
  digitalWrite(RST, HIGH);

  Serial.begin(115200);
  while (!Serial) {
    delay(1);
  }
  delay(100);
  Serial.println("LoRa Tx Node");

  // manual reset
  digitalWrite(RST, LOW);
  delay(10);
  digitalWrite(RST, HIGH);
  delay(10);

  while (!rf95.init()) {
    Serial.println("Initialization Failed!");
    while (1);
  }
  Serial.println("LoRa Initialized!");

  if (!rf95.setFrequency(FREQ)) {
    Serial.println("setFrequency failed");
    while (1);
  }
  Serial.print("Frequency set to: ");
Serial.println(FREQ);
  rf95.setTxPower(23, false);
}
```

Tx Program: LoRa interfaced with NodeMCU

```
void loop()
{
  //Reading data from the DHT sensor
  hum = dht.readHumidity();
  temp= dht.readTemperature();
  String msg1= "Temp: ";
  msg1 += temp;
  msg1 += "C, Hum: ";
  msg1 += hum;
  msg1 += "%";
  delay(1000); // Delay of 1 second before
transmitting the data
  Serial.println("Sending temperature and
humidity");

 //Send data to the receiver
  char radiopacket[26];
  msg1.toCharArray(radiopacket,26);
  Serial.println(radiopacket);
  delay(10);
  rf95.send((uint8_t *)radiopacket, 26);
  delay(10);
  rf95.waitPacketSent();
  uint8_t buf[RH_RF95_MAX_MESSAGE_LEN];
  uint8_t len = sizeof(buf);

 if (rf95.waitAvailableTimeout(1000))
  {
   if (rf95.recv(buf, &len))
   {
     Serial.print("Acknowledgement
Received!\n");
    }
    else
    {
     Serial.println("Receive failed\n");
    }
  }
  else
  {
   Serial.println("No Receiver Node Found!");
  }
}
```

Rx Program: LoRa interfaced with NodeMCU

```
#include <SPI.h>
#include <RH_RF95.h>

#define CS  2    // "E"
#define RST 5   // "D"
#define INT 15   // "B"

#define FREQ 915.0

RH_RF95 rf95(CS, INT);

#define LED 4 //GPIO4- D2

void setup()
{
  pinMode(LED, OUTPUT);
  pinMode(RST, OUTPUT);
  digitalWrite(RST, HIGH);
Serial.begin(115200);
while (!Serial) {
  delay(1);
}
delay(100);

Serial.println("LoRa Rx Node");
digitalWrite(RST, LOW);//Reset manually
delay(10);
digitalWrite(RST, HIGH);
delay(10);

while (!rf95.init()) {
  Serial.println("Initialization Failed!");
  while (1);
}
Serial.println("LoRa Initialized!");

if (!rf95.setFrequency(FREQ)) {
    Serial.println("setFrequency failed");
    while (1);
  }
  Serial.print("Frequency set to: ");
Serial.println(FREQ);

  rf95.setTxPower(23, false);
}
```

Rx Program: LoRa interfaced with NodeMCU

```
void loop()
{
  if (rf95.available())
  {
    uint8_t
buf[RH_RF95_MAX_MESSAGE_LEN];
    uint8_t len = sizeof(buf);

    if (rf95.recv(buf, &len))
    {
      digitalWrite(LED, HIGH);
      //RH_RF95::printBuffer("Received: ", buf,
len);
      Serial.print("Received: ");
      Serial.println((char*)buf);

      // Send a reply
      uint8_t data[] = "Data Received";
      rf95.send(data, sizeof(data));
      rf95.waitPacketSent();
      Serial.println("Acknowledged!");
      digitalWrite(LED, LOW);
    }
    else
    {
      Serial.println("Receive failed");
    }
  }
}
```

Thank You!!

Key Enablers of Industrial IoT: Connectivity – Part 5

Dr. Sudip Misra
Professor
Department of Computer Science and Engineering
Indian Institute of Technology Kharagpur
Email: smisra@sit.iitkgp.ernet.in
Website: http://cse.iitkgp.ac.in/~smisra/
Research Lab: cse.iitkgp.ac.in/~smisra/swan/

Hands-On
(Zigbee Connectivity)

System Overview

- Basic connectivity model to enable data transfer between xbee modules is discussed. The hands-on focuses on the following areas:
  - Basic configuration of Xbee module
  - Introduction to basic communication between two Xbee modules using python programming language.

Zigbee

Introduction to Zigbee

- Zigbee is a communication protocol with its physical and MAC layer based on the IEEE 802.15.4.
- It is one of the well known standards for low power low data rate WPAN.
- Zigbee supports 3 topologies: Start, Tree and Mesh
- It is mostly used in home and industrial automation applications.
- The communication ranges varies between 10-100 meters depending on the device variant.

Introduction to Zigbee (Contd.)

- A Zigbee device can be any of the three types: 1) Coordinator 2) Router and 3) End device.
- A coordinator is the root of a the network and acts as a bridge between different networks.
- Router relays the information to other nodes in the network. It can also run small scale applications
- End devices are only responsible to connect to the parent node, no relaying of information is supported.

Source: Tarun Agarwal, ZigBee Wireless Technology Architecture and Applications

Zigbee and Xbee

- Zigbee is a mesh communication protocol based on the IEEE 802.15.4
- Xbee is the product that uses the Zigbee communication protocol for radio communication.
- Xbee is a product by Digi which comes in may variants.
- Digimesh is another protocol that works similar to Zigbee with additional desirable features.

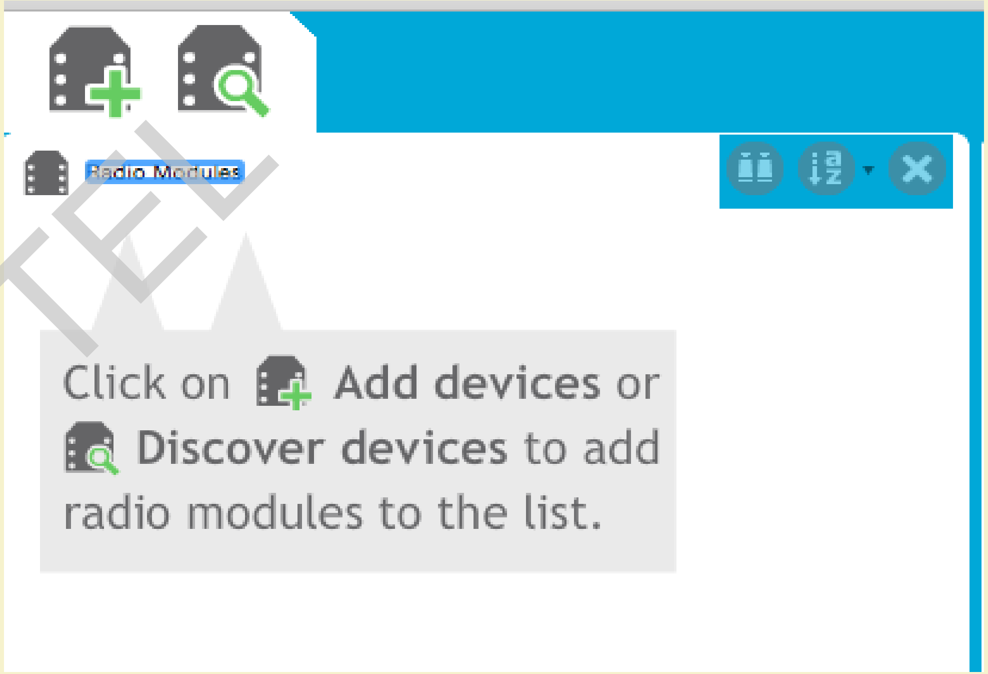
Source: ZigBee Vs. XBee: An Easy-To-Understand Comparison

Pre-requisites

- Install the xbee library
  - Pip install xbee
- Install XCTU software from here.
- XCTU will be used to configure the xbee modules before using them for communication.

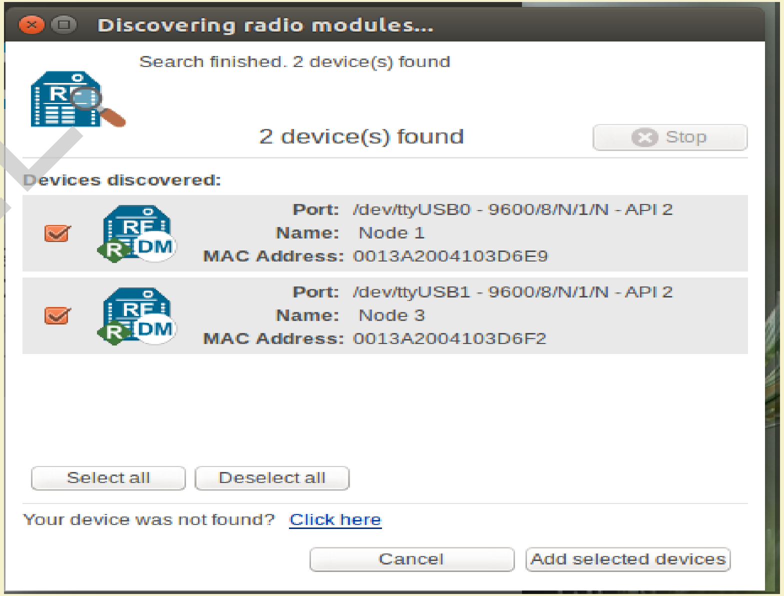
Xbee Configuration

- Open XCTU.
- Click on the discover button to discover the Xbee devices which are currently connected in the COM ports.



Xbee Configuration (cont.)

- After discovering the devices, identify the port id and the MAC address of the Xbee devices.
- Port id and MAC id are required for the communication.



Tx Program: Xbee Transmitter

```
from xbee import DigiMesh
import time
import serial
PORT = '/dev/ttyUSB1' #sender port id
BAUDRATE = 9600
ser = serial.Serial(PORT,BAUDRATE)
def send(ser, msg, addr64='000000000000FFFF'):
    xbee = DigiMesh(ser,escaped=True)
    if(ser.isOpen()==False) :
        ser.open()
    addr64 = bytearray.fromhex(addr64)
    xbee.tx(
        frame_id = b'\x00',
        dest_addr = addr64,
        data = msg.encode('utf8')
    )
msg=raw_input("Enter message:")
send(ser,msg)
```

- Importing the library files of DigiMesh protocol.
- Sender port id.
- dest_addr refers to destination address. The default target is to broadcast the message.

Rx Program: Xbee Receiver

```
from xbee import DigiMesh
import time
import serial
PORT = '/dev/ttyUSB0'
BAUDRATE = 9600
ser = serial.Serial(PORT,BAUDRATE)
xbee = DigiMesh(ser,escaped=True)
while True:
    try:
        response = xbee.wait_read_frame()
        if response['id']=='rx' :
            print(response['data'].decode('utf8'),)
    except KeyboardInterrupt:
        ser.close()
        break
```

- Importing the library files of DigiMesh protocol.
- Receiver port id.
- Waiting for receiving the data from sender.

Output Console for Transmitter

```
swan1@swan1-Inspiron-660s:~/XBEE_DEMO$ python sender.py
Enter message:Welcome to IIOT course
```

Output Console for Receiver

```
swan1@swan1-Inspiron-660s:~/XBEE_DEMO$ python receiver.py
(u'Welcome to IIOT course',)
```

References

1. XCTU: Next Generation Configuration Platform for XBee/RF Solutions. Online. https://www.digi.com/products/xbee-rf-solutions/xctu-software/xctu#productsupport-utilities
2. Tarun Agarwal, ZigBee Wireless Technology Architecture and Applications. Online. URL: https://www.elprocus.com/what-is-zigbee-technology-architecture-and-its-applications/
3. Xbee. Online. URL: https://pypi.org/project/XBee/
4. Glenn Schatz**.** April 15, 2016. ZigBee Vs. XBee: An Easy-To-Understand Comparison. Online. URL: https://www.link-labs.com/blog/zigbee-vs-xbee

Thank You!!

IIT KHARAGPUR

NPTEL ONLINE CERTIFICATION COURSES

NPTEL

Key Enablers of Industrial IoT: Processing-Part 1

Dr. Sudip Misra

Professor

Department of Computer Science and Engineering

Indian Institute of Technology Kharagpur

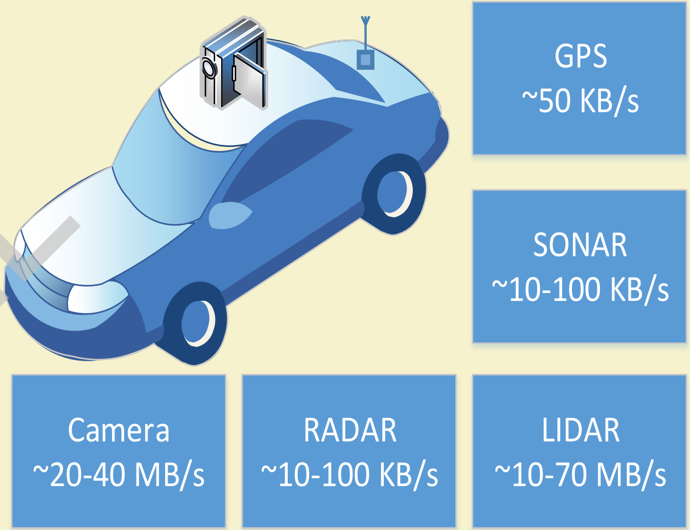
Email: smisra@sit.iitkgp.ernet.in

Website: http://cse.iitkgp.ac.in/~smisra/

Research Lab: cse.iitkgp.ac.in/~smisra/swan/

IIoT Processing: Necessity

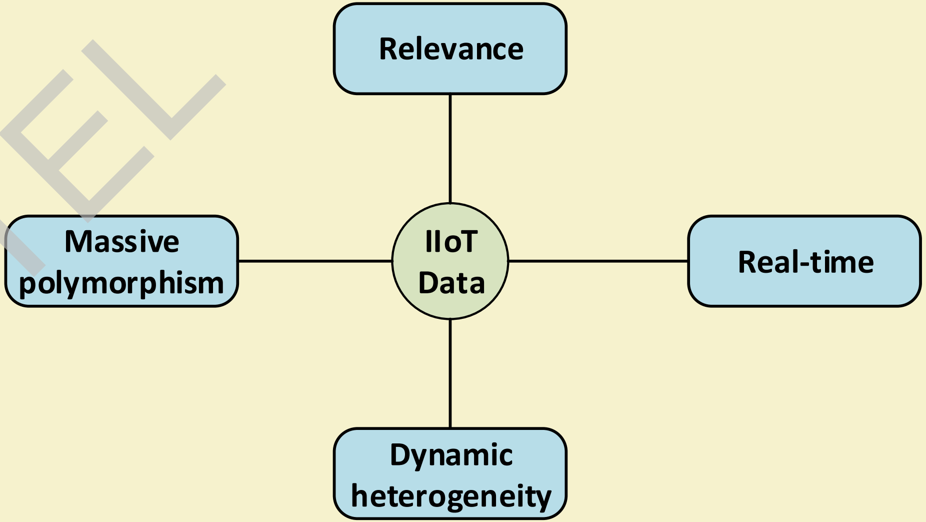
- Billions of connected devices
  - Cisco prediction of 50 billion connected devices by 2020
  - Autonomous cars generate ~100 MB data per second
  - Intermittent, unstructured, highly diverse data
  - Businesses do not need raw data deluge; need *insights* from data in real-time



Source: Self driving cars, Intel

IIoT Processing: Data characteristics

- Polymorphism
  - Heterogeneous sensors – pressure, vibration, sound
  - Different metrics, precision, formats
- Temporal/causal relationships in data
- Correlation in space, time and other dimensions



IIoT Processing: Challenges

- Complexity of data is increasing
  - Cyber Physical Systems (CPS)
  - Distributed connected applications
    - Need to interpret patterns
    - Accurate decisions with minimal latency
  - Analysis before storage
- Complex Event Processing (CEP)
  - Analyse and correlate event streams from different data sources

IIoT Processing: Complex Event processing (CEP)

- Rule-based engine
  - Extract causal and temporal patterns using predefined rules
  - Handles multiple data streams and correlates them to provide meaningful output
  - Can process data in near real-time

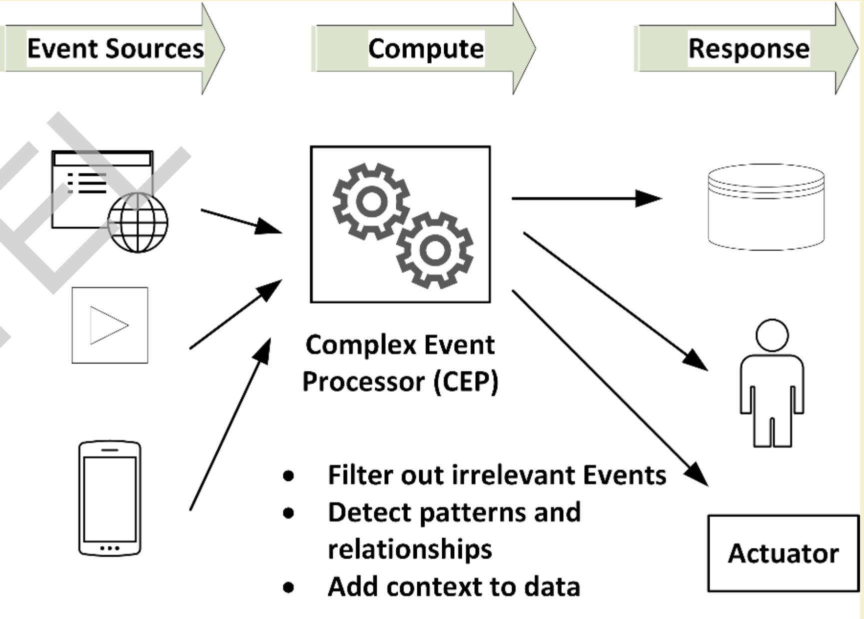
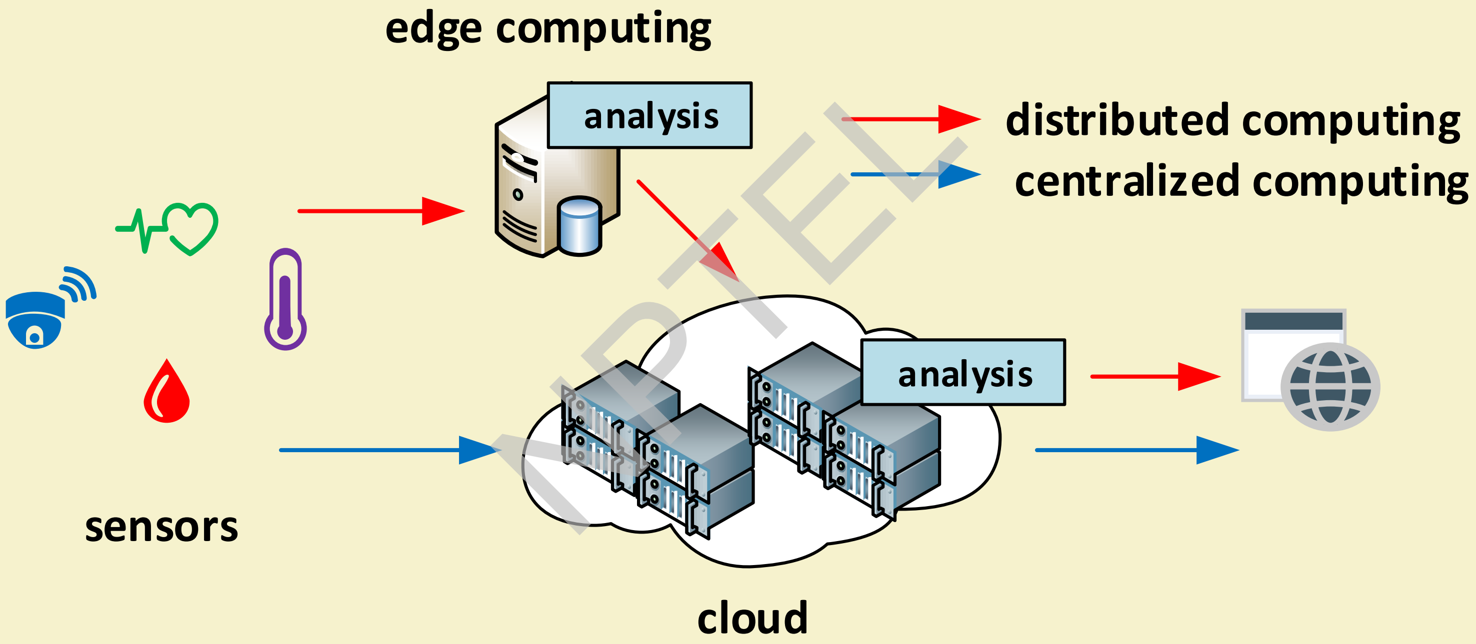


Figure: CEP Components

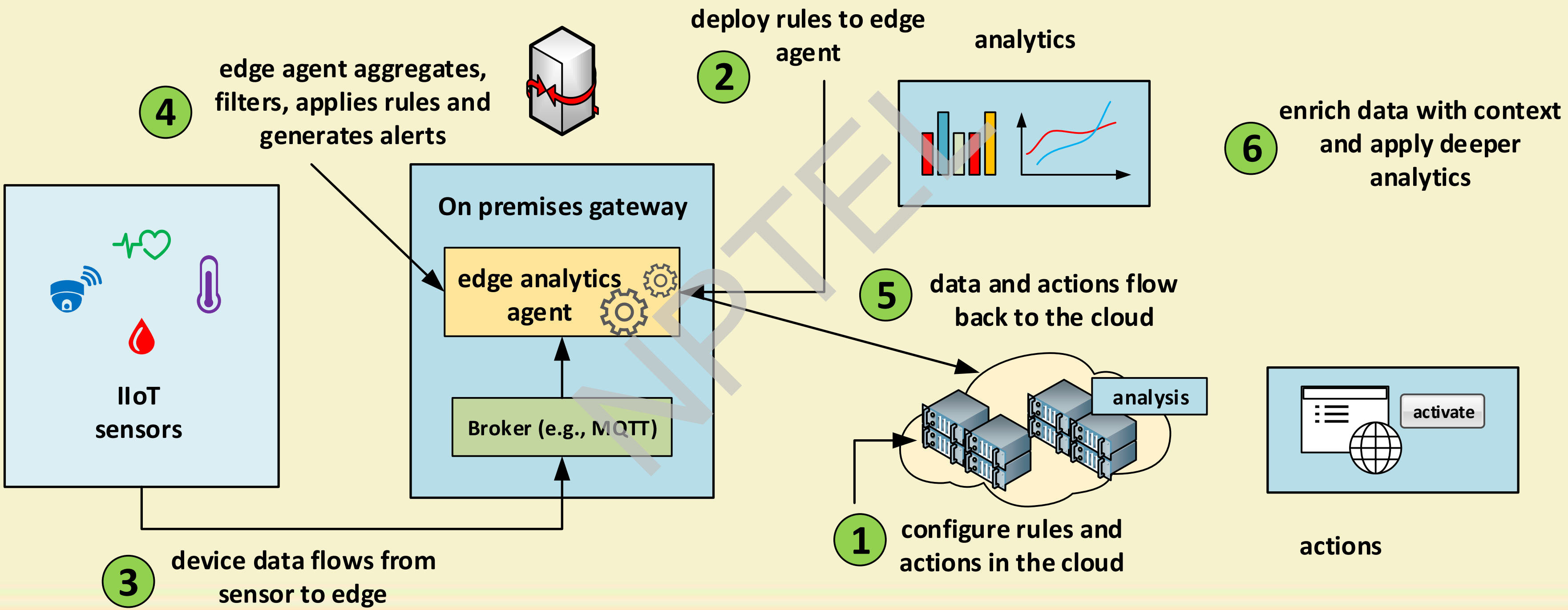
IIoT Processing: Types



IIoT Processing: Middleware

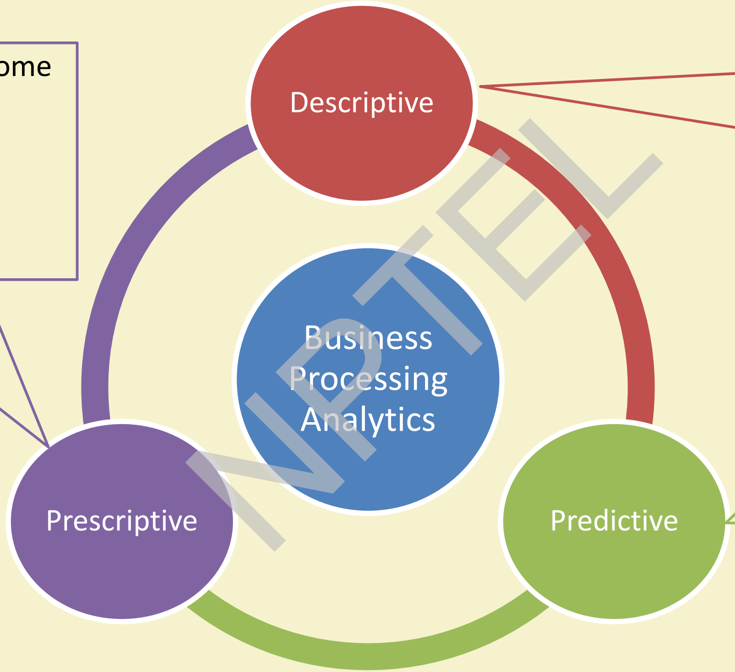
- Software layer between infrastructure layer and application layer
  - Provides services according to device functionality
  - Support for heterogeneity, security
  - Many middleware solutions are based on service-oriented architecture (SOA)

IIoT Processing: End to End



IIoT: Processing & Analytics

Questions: What & why to perform some action
Enablers: Optimization/Simulation/ Decision models
Outcomes: Best possible business decision



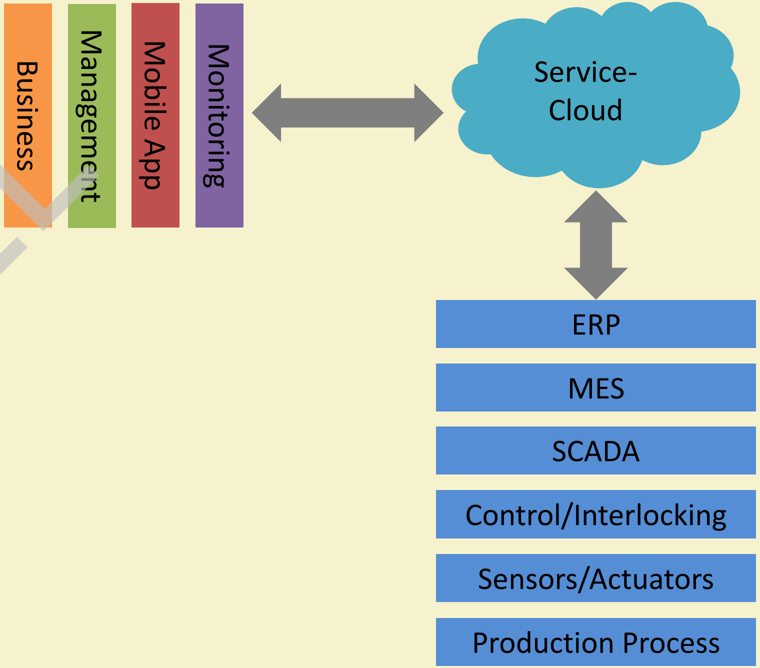
Questions: What already happened and currently happening
Enablers: Dashboard/Reports/Scorecards
Outcomes: Business questions & opportunities

Questions: What will happen and why
Enablers: Data mining/Web mining/ predictions
Outcomes: Forecasting of future conditions

Source: Wang *et al.*, TENCON 2015

IIoT Processing: Supervisory Control & Management

- *Challenge*: Management of the huge number of heterogeneous devices in the SOA-based collaboration
- *Function*: Dynamic control & automation as per the business requirements
- Service-Cloud
  - Facilitates the remote supervisory control
  - Dynamic & rapid composition of multiple services
  - Virtualization of the automation hierarchy



Source: Colombo *et al.*, Springer 2014

MIDAS: IoT/M2M Platforms

- Modular, scalable & secure architecture
- Flexible design – facility for both on premise and cloud-based deployment
- Reliable data transfer with support for many existing protocols
- Provide a platform for custom application design
- Analytics platform:
  - Both runtime and batch analytics
  - Repository consists of pre-designed solutions

Source: MIDAS: IoT/M2M Platforms

IIoT Processing: On-going Research

- Content-aware processing
- Analytical energy model of IIoT
  - Relationship between transmission and processing energy costs
  - Exact expression of stochastic fluid model relating data correlation coefficient and computing types
- Results
  - Distributed computing is applicable for highly correlated data sources

Source: Zhou et al., 2018

IIoT Processing: On-going Research (cont.)

- <u>Context-aware</u> stream processing
- Limitation of current CEP systems
  - <u>Manual</u> threshold specification
  - Run-time update of threshold not possible
  - Not context-aware
- Proposed uCEP engine
  - Uses <u>adaptive clustering</u> techniques to dynamically detect boundaries between CEP values and find optimal rules
  - Extract causal and temporal patterns using <u>adaptive rules</u>

Source: Akbar et al., 2015

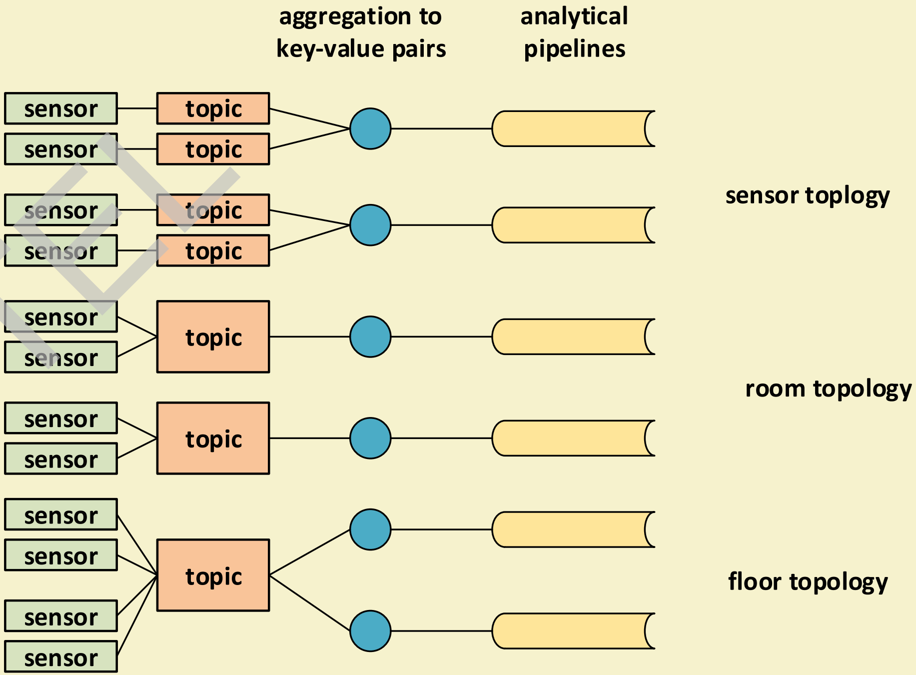
IIoT Processing: On-going Research (cont.)

- Processing topologies
  - Real-time IoT processing systems use message brokers (e.g. MQTT, Apache Kafka) and transfer them to analytical pipelines
  - Single message queue – not scalable, increased latency
  - Size of queue increases with increase in
    - Data volume
    - Number of sensors
    - Out of order data that needs more buffer space
  - Naive approach – Install more servers
    - Impractical
    - Existing server not fully utilized

Source: Dey et al., 2015

IIoT Processing: On-going Research (cont.)

- Producer phase
  - Similar across topologies
- Consumer phase
  - Extracts topic data and converts into key-value pairs
  - Workload increases from sensor to floor topology
- Modelling phase
  - Workload of room topology is reduced compared to sensor topology



Source: Dey et al., 2015

IIoT Processing: On-going Research (cont.)

- Semantic Rules Engine (SRE)
  - Rules Engine deployed at the gateways
    - high level concepts such as location and measurement type used for rule formation
  - Semantic engine to provide abstraction heterogeneity of devices
    - Business logic automatically implemented as low level rules
  - Leverage device metadata and enable retrieval of contextual data from devices

Source: Kaed et al., 2018

IIoT Processing: On-going Research (cont.)

- Big data analytics for maritime industry (Wang *et al.*, TENCON 2015)
  - Two-layer BDA-IIoT framework
    - Vessel BDA+IIoT
      - On-board, real-time & local processing
      - Limited resources
      - IIoT: Consists of *communication technologies, sensor/actuators, devices/machinery*
      - Vessel BDA: *CPU/GPU, Storage, Virtualization*
    - Land BDA
      - Remote high-power computing
      - Components: *CPU/GPU/ Cloud, Storage, Virtualization*

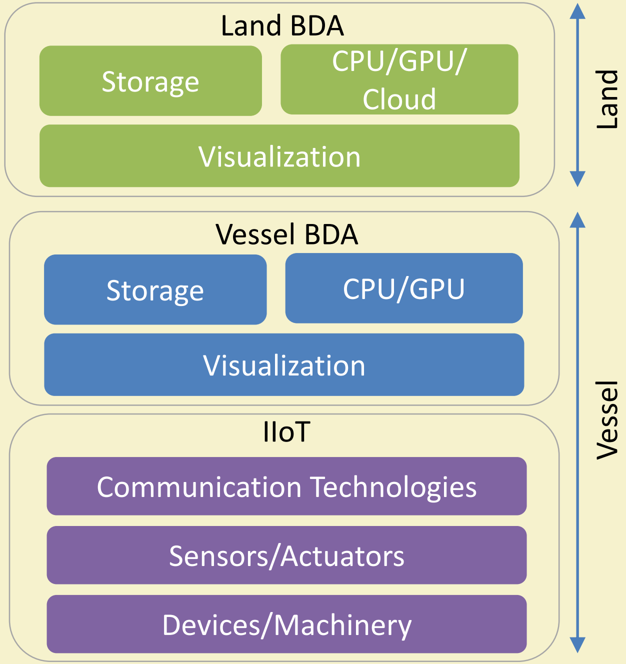
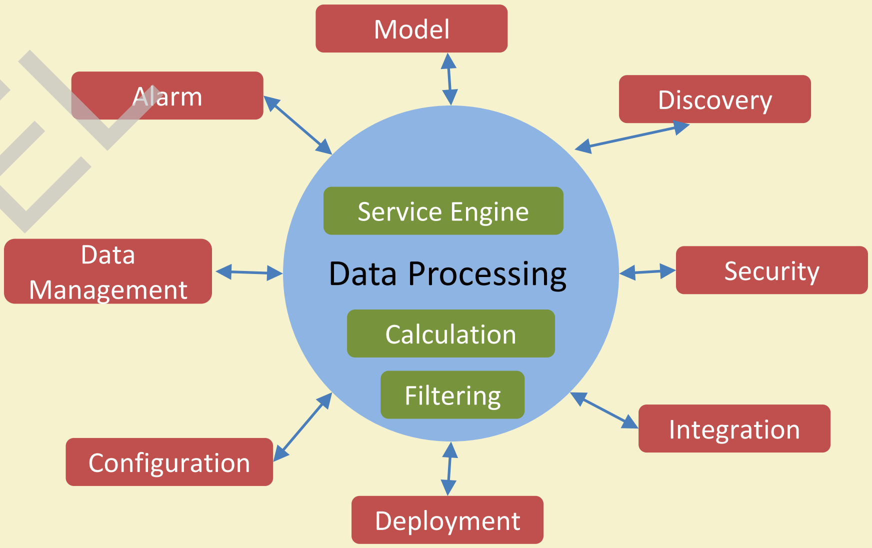


Fig: BDA-IIoT Framework

Source: Wang et al., 2015

IIoT Processing: On-going Research (cont.)

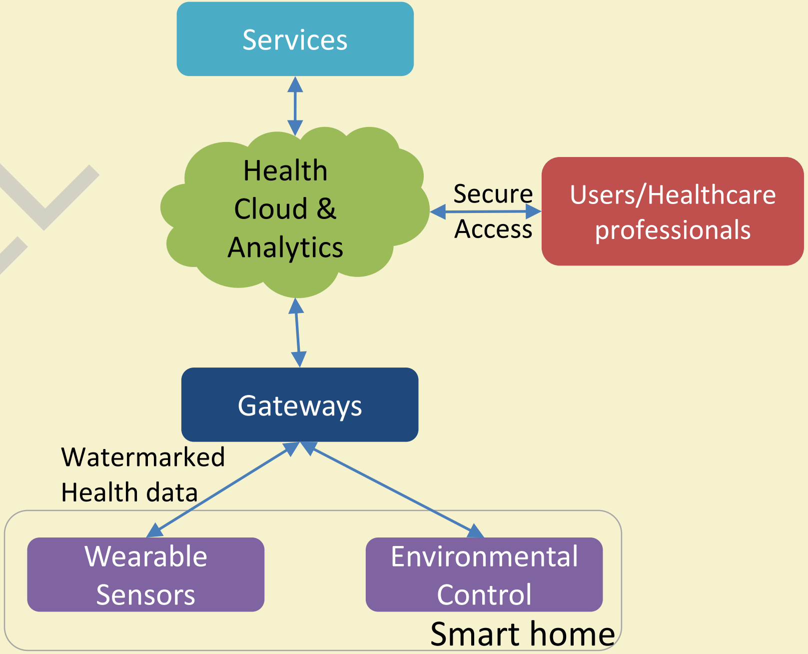
- Data Processing [Karnouskos *et al.*, 2014]
  - <u>Functional group & block</u>: In devices or in cloud
  - <u>Services</u>: Simple filtering to complex analytics
  - <u>*Complex event processing (CEP)*</u>: Real-time correlation & aggregation of event data
  - <u>Rule-based deployment</u> on incoming events
  - <u>API-based facility to create, modify, or delete rules</u>



Source: Karnouskos et al., 2014

IIoT Processing: On-going Research (cont.)

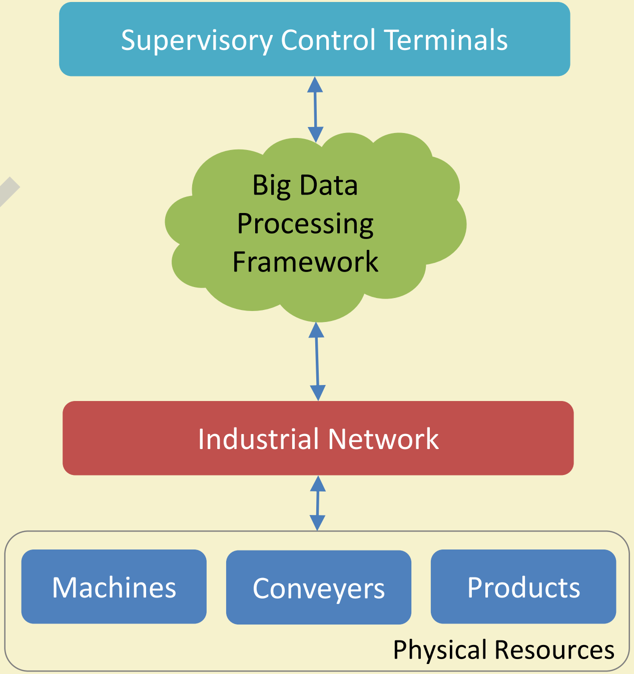
- HealthIIoT [Hossain et al., 2016]
  - Health data collected by sensor-equipped wearable devices
  - Cloud-based analytics for <u>clinical prediction</u>
  - Incorporates *watermarking* & *user identification* in the health data to enhance security
  - Cloud-based dynamic resource management & service provisioning
  - <u>Health condition monitoring</u> by in-loop healthcare professional



Source: Hossain et al., 2016

IIoT Processing: On-going Research (cont.)

- Self-organized Multi-agent System in Smart Factory [Wang et al., 2016]
  - Components: *cloud, industrial network, smart terminals*
  - Increased flexibility due to distributed cooperation and autonomous decision making framework
  - Self-organizing is achieved by intelligent negotiations between agents
  - Cloud-based big data processing framework assists the self-organization & supervisory control



Source: Wang et al., 2016

IIoT Processing: On-going Research (cont.)

- Line Information System Architecture (LISA) [Theorin et al. , 2017]
  - Event-driven information system
  - Loosely-coupled system with prototype-oriented information model
  - Components
    - *LISA events*: machine state change, occurrence of new information
    - *Message bus*: enterprise service bus with standard & structured framework for message routing
    - *Communication end-points*: interoperable communication for services
    - *Service end-points*: interoperable communication to standard interfaces

Source: Theorin et al., 2017

References

[1] A. Dey, K. Stuart and M. E. Tolentino, “Characterizing the impact of topology on IoT stream processing,” *in Proc. of the IEEE World Forum on Internet of Things (WF-IoT)*, 2018, pp. 505-510.

[2] C. E. Kaed, I. Khan, A. Van Den Berg, H. Hossayni and C. Saint-Marcel, “SRE: Semantic Rules Engine for the Industrial Internet-Of-Things Gateways,” in *IEEE Transactions on Industrial Informatics*, vol. 14, no. 2, pp. 715-724, 2018.

[3] A. Akbar, F. Carrez, K. Moessner, J. Sancho and J. Rico, “Context-aware stream processing for distributed IoT applications," *in Proc. of the IEEE World Forum on Internet of Things (WF-IoT)*, 2015, pp. 663-668.

[4] L. Zhou, D. Wu, J. Chen and Z. Dong, “When Computation Hugs Intelligence: Content-Aware Data Processing for Industrial IoT," in *IEEE Internet of Things Journal*, vol. 5, no. 3, pp. 1657-1666, 2018.

[5] H. Wang, O. L. Osen, G. Lit, W. Lit , H.-N. Dai, W. Zeng, “Big Data and Industrial Internet of Things for the Maritime Industry in Northwestern Norway,” in *Proc. IEEE TENCON*, Macao, China, 2015.

[6] A. W. Colombo, S. Karnouskos and T. Bangemann, “Towards the Next Generation of Industrial Cyber-Physical Systems,”*Industrial Cloud-Based Cyber-Physical Systems*, A. W. Colombo et al. (eds.), Springer, 2014.

References

[7] S. Karnouskos, A. W. Colombo, T. Bangemann, K. Manninen, R. Camp, M. Tilly, M. Sikora, F. Jammes, J. Delsing, J. Eliasson, P. Nappey, J. Hu and M. Graf, "The IMC-AESOP Architecture for Cloud-Based Industrial Cyber-Physical Systems,"*Industrial Cloud-Based Cyber-Physical Systems*, A. W. Colombo et al. (eds.), Springer, 2014.

[8] M. S. Hossain and G. Muhammad, "Cloud-assisted Industrial Internet of Things (IIoT) – Enabled framework for health monitoring," *Computer Networks*, vol. 101, pp. 192-202, 2016.

[9] S. Wang, J. Wan, D. Zhang, D. Li, C. Zhang, "Towards smart factory for industry 4.0: a self-organized multi-agent system with big data based feedback and coordination," Computer Networks, vol. 101, pp. 158-168, 2016.

[10] A. Theorin, K. Bengtsson, J. Provost, M. Lieder, C. Johnsson, T. Lundholm, and B. Lennartson, "An event-driven manufacturing information system architecture for Industry 4.0," International Journal of Production Research, vol 55, no. 5, pp. 1297-1311, 2017.

[11] MIDAS: IoT/M2M Platforms, Web: https://www.happiestminds.com/solutions/iot-service-platform-midas/

Thank You!!

IIT KHARAGPUR

NPTEL ONLINE CERTIFICATION COURSES

NPTEL

Key Enablers of Industrial IoT: Processing-Part 1

Dr. Sudip Misra

Professor

Department of Computer Science and Engineering

Indian Institute of Technology Kharagpur

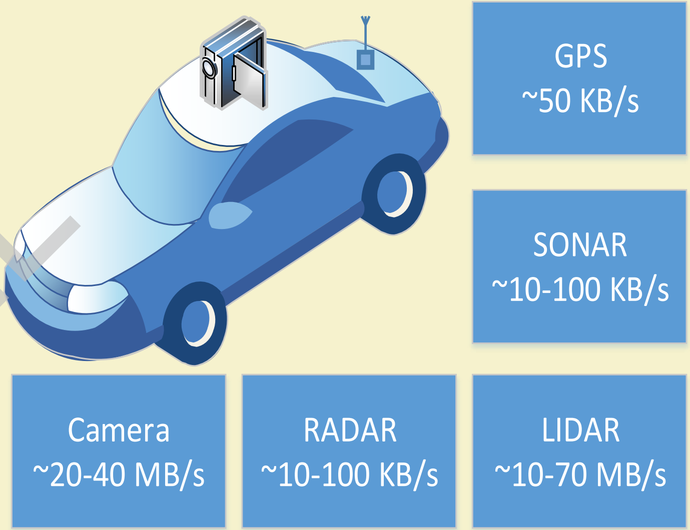
Email: smisra@sit.iitkgp.ernet.in

Website: http://cse.iitkgp.ac.in/~smisra/

Research Lab: cse.iitkgp.ac.in/~smisra/swan/

IIoT Processing: Necessity

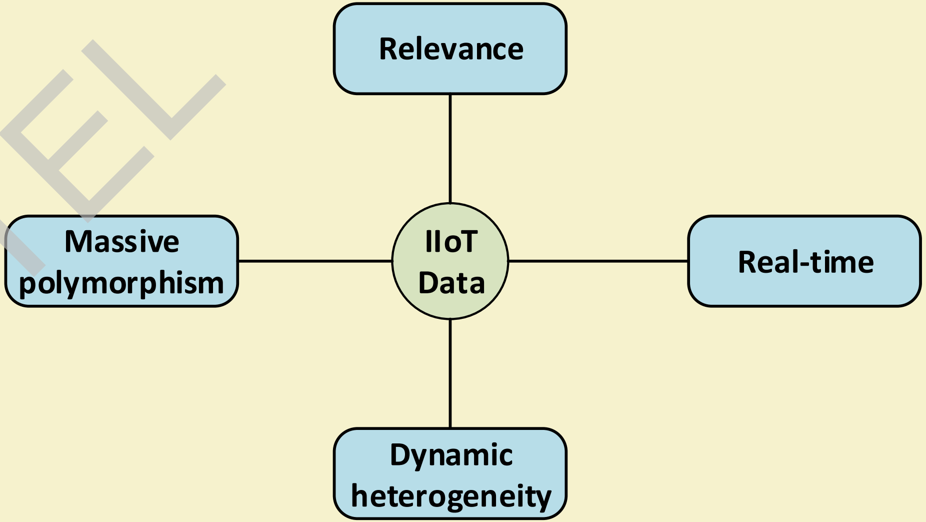
- Billions of connected devices
  - Cisco prediction of 50 billion connected devices by 2020
  - Autonomous cars generate ~100 MB data per second
  - Intermittent, unstructured, highly diverse data
  - Businesses do not need raw data deluge; need *insights* from data in real-time



Source: Self driving cars, Intel

IIoT Processing: Data characteristics

- Polymorphism
  - Heterogeneous sensors – pressure, vibration, sound
  - Different metrics, precision, formats
- Temporal/causal relationships in data
- Correlation in space, time and other dimensions



IIoT Processing: Challenges

- Complexity of data is increasing
  - Cyber Physical Systems (CPS)
  - Distributed connected applications
    - Need to <u>interpret patterns</u>
    - <u>Accurate decisions</u> with <u>minimal latency</u>
  - Analysis before storage
- Complex Event Processing (CEP)
  - <u>Analyse and correlate event streams</u> from different data sources

IIoT Processing: Complex Event processing (CEP)

- Rule-based engine
  - Extract causal and temporal patterns using predefined rules
  - Handles multiple data streams and correlates them to provide meaningful output
  - Can process data in near real-time

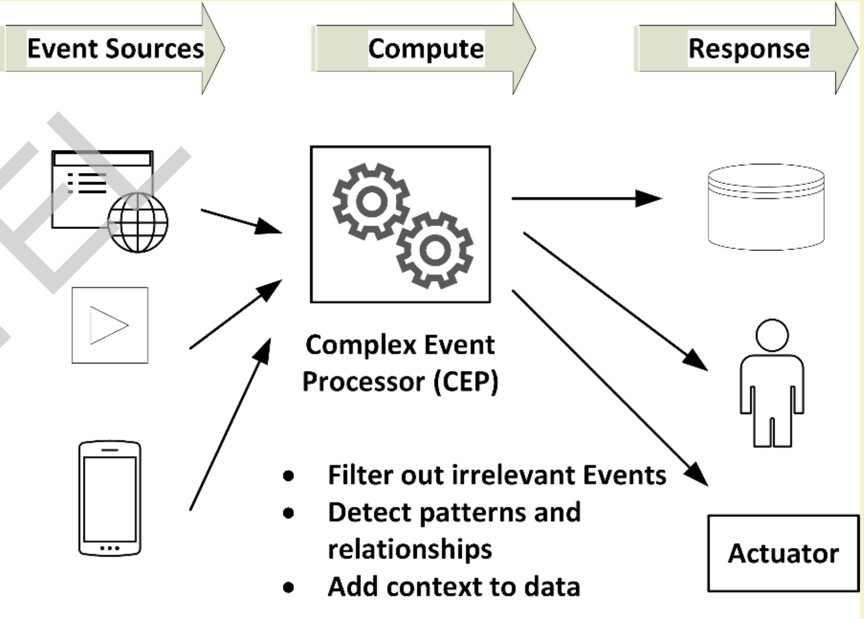
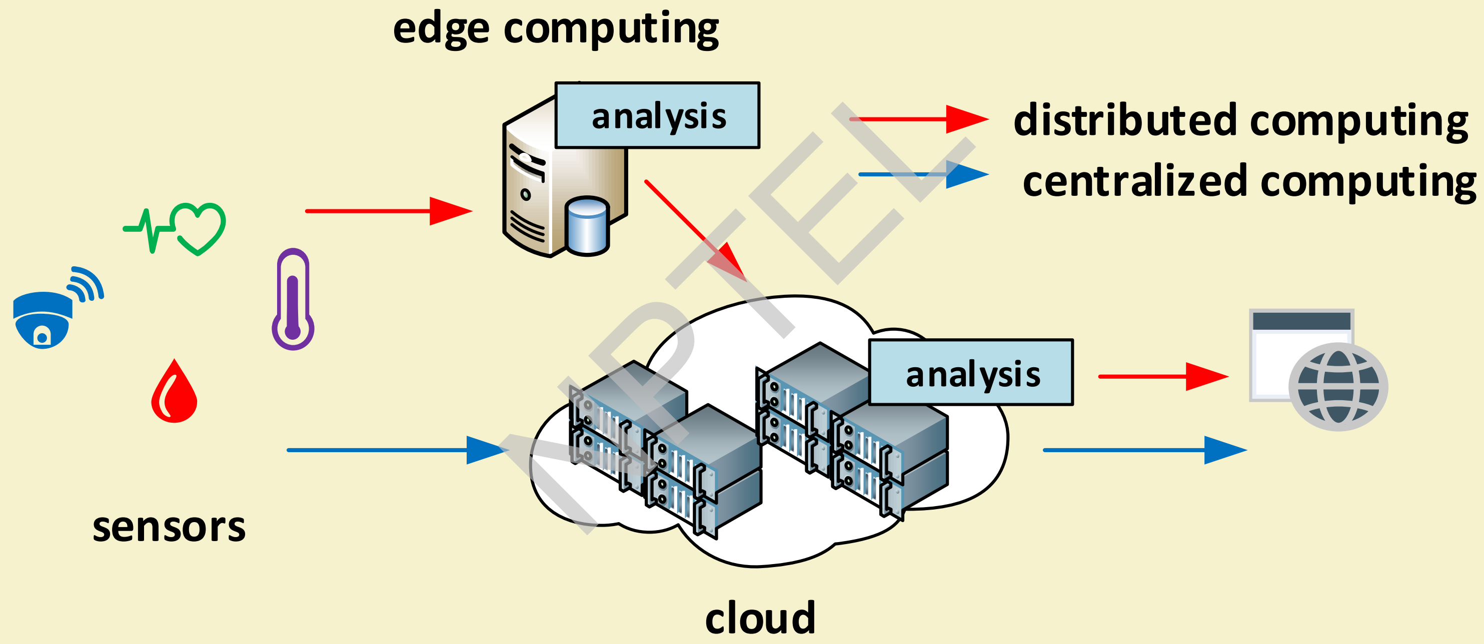


Figure: CEP Components

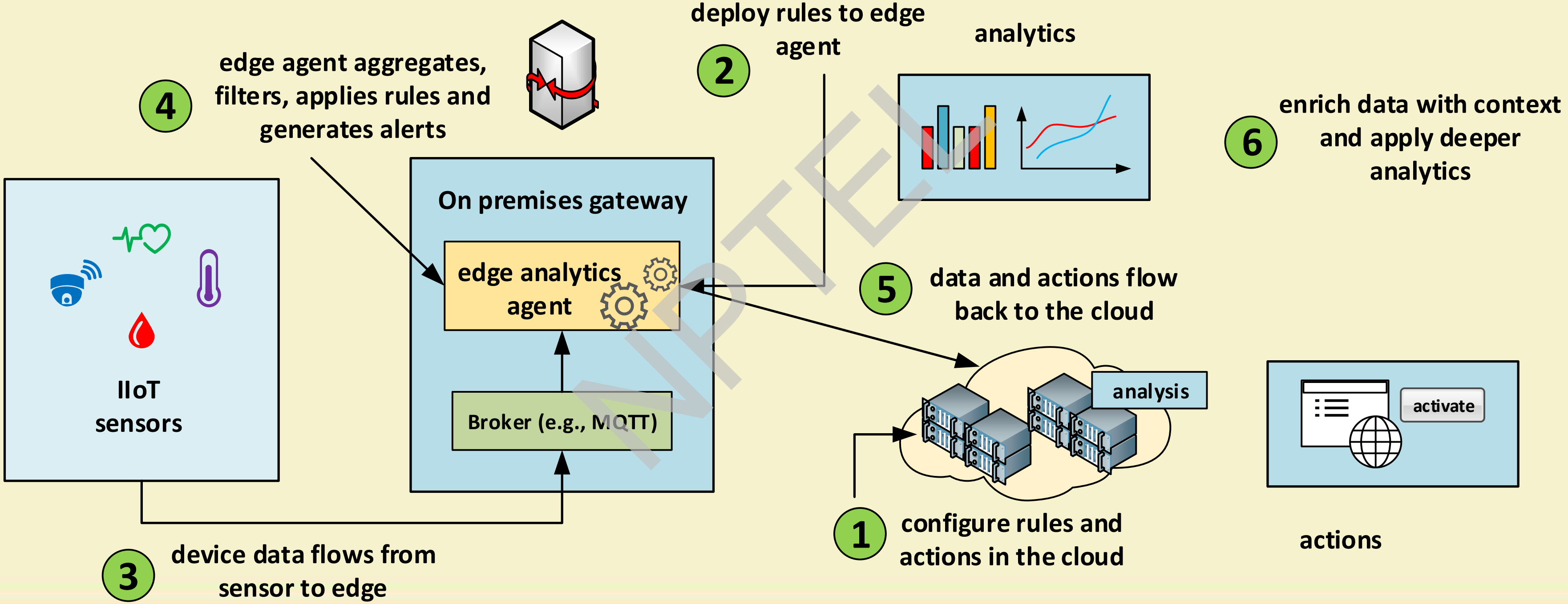
IIoT Processing: Types



IIoT Processing: Middleware

- Software layer between infrastructure layer and application layer
  - Provides services according to device functionality
  - Support for heterogeneity, security
  - Many middleware solutions are based on service-oriented architecture (SOA)

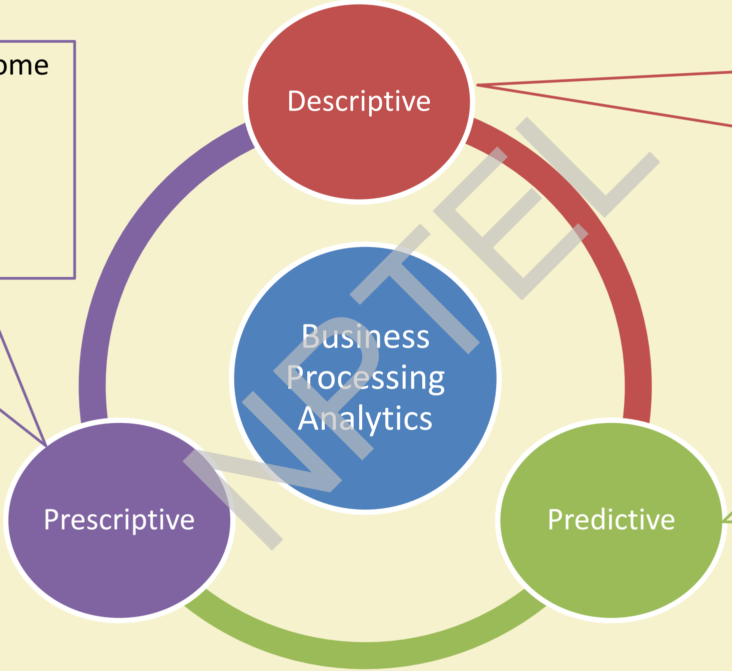
IIoT Processing: End to End



IIoT: Processing & Analytics

Questions: What already happened and currently happening
Enablers: Dashboard/Reports/Scorecards
Outcomes: Business questions & opportunities

Questions: What & why to perform some action
Enablers: Optimization/Simulation/ Decision models
Outcomes: Best possible business decision

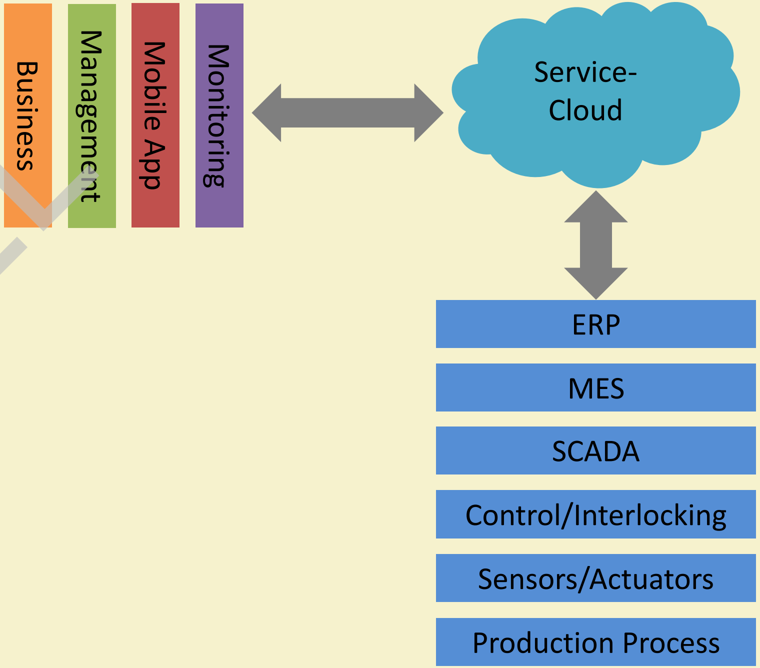


Questions: What will happen and why
Enablers: Data mining/Web mining/ predictions
Outcomes: Forecasting of future conditions

Source: Wang *et al*., TENCON 2015

IIoT Processing: Supervisory Control & Management

- *Challenge*: Management of the huge number of heterogeneous devices in the SOA-based collaboration
- *Function*: Dynamic control & automation as per the business requirements
- Service-Cloud
  - Facilitates the <u>remote supervisory control</u>
  - Dynamic & rapid <u>composition of multiple services</u>
  - <u>Virtualization of the automation hierarchy</u>



Source: Colombo *et al.*, Springer 2014

MIDAS: IoT/M2M Platforms

- Modular, scalable & secure architecture
- Flexible design – facility for both on premise and cloud-based deployment
- Reliable data transfer with support for many existing protocols
- Provide a platform for custom application design
- Analytics platform:
  - Both runtime and batch analytics
  - Repository consists of pre-designed solutions

Source: MIDAS: IoT/M2M Platforms

IIoT Processing: On-going Research

- Content-aware processing
- Analytical energy model of IIoT
  - Relationship between transmission and processing energy costs
  - Exact expression of stochastic fluid model relating data correlation coefficient and computing types
- Results
  - Distributed computing is applicable for highly correlated data sources

Source: Zhou et al., 2018

IIoT Processing: On-going Research (cont.)

- Context-aware stream processing
- Limitation of current CEP systems
  - Manual threshold specification
  - Run-time update of threshold not possible
  - Not context-aware
- Proposed uCEP engine
  - Uses adaptive clustering techniques to dynamically detect boundaries between CEP values and find optimal rules
  - Extract causal and temporal patterns using adaptive rules

Source: Akbar et al., 2015

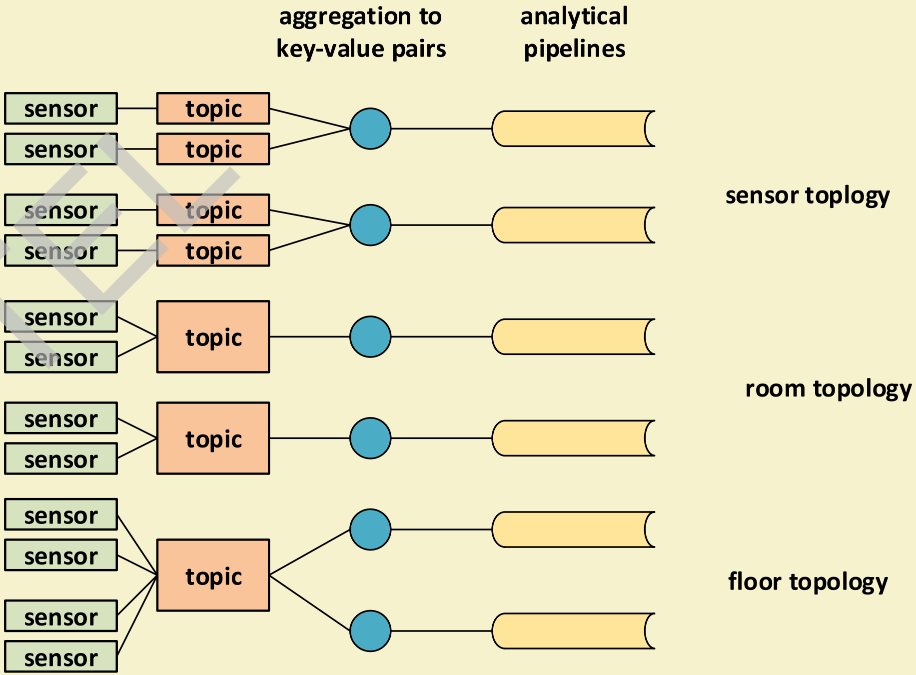
IIoT Processing: On-going Research (cont.)

- Processing topologies
  - Real-time IoT processing systems use message brokers (e.g. MQTT, Apache Kafka) and transfer them to analytical pipelines
  - Single message queue – not scalable, increased latency
  - Size of queue increases with increase in
    - Data volume
    - Number of sensors
    - Out of order data that needs more buffer space
  - Naive approach – Install more servers
    - Impractical
    - Existing server not fully utilized

Source: Dey et al., 2015

IIoT Processing: On-going Research (cont.)

- Producer phase
  - Similar across topologies
- Consumer phase
  - Extracts topic data and converts into key-value pairs
  - Workload increases from sensor to floor topology
- Modelling phase
  - Workload of room topology is reduced compared to sensor topology



Source: Dey et al., 2015

IIoT Processing: On-going Research (cont.)

- Semantic Rules Engine (SRE)
  - Rules Engine deployed at the gateways
    - high level concepts such as location and measurement type used for rule formation
  - Semantic engine to provide abstraction heterogeneity of devices
    - Business logic automatically implemented as low level rules
  - Leverage device metadata and enable retrieval of contextual data from devices

Source: Kaed et al., 2018

IIoT Processing: On-going Research (cont.)

- Big data analytics for maritime industry (Wang *et al.*, TENCON 2015)
  - Two-layer BDA-IIoT framework
    - Vessel BDA+IIoT
      - On-board, real-time & local processing
      - Limited resources
      - IIoT: Consists of *communication technologies, sensor/actuators, devices/machinery*
      - Vessel BDA: *CPU/GPU, Storage, Virtualization*
    - Land BDA
      - Remote high-power computing
      - Components: *CPU/GPU/ Cloud, Storage, Virtualization*

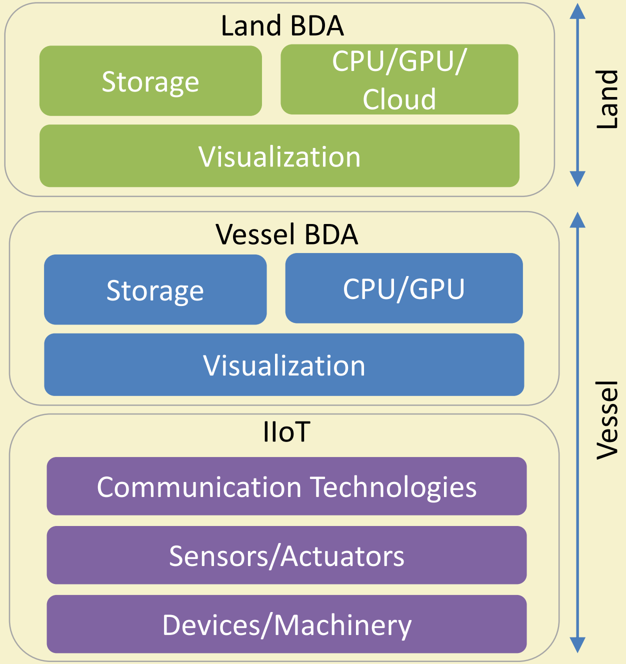
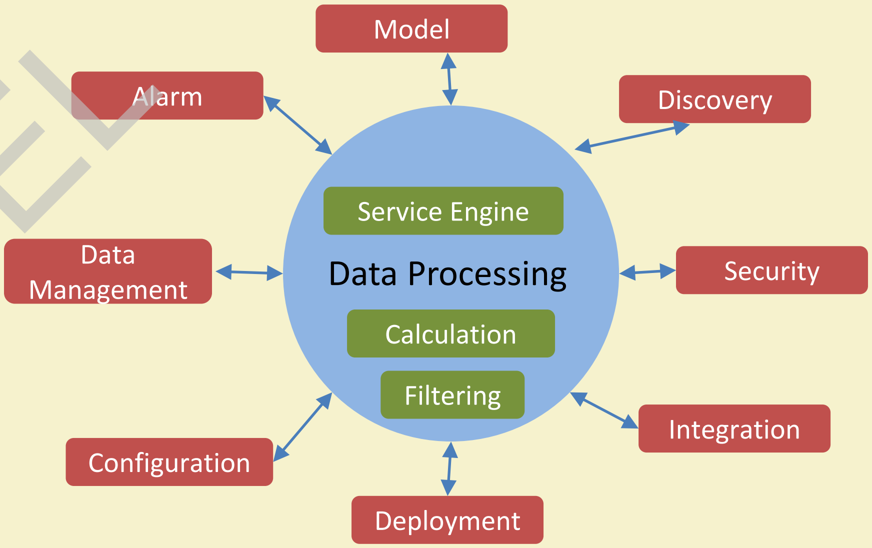


Fig: BDA-IIoT Framework

Source: Wang et al., 2015

IIoT Processing: On-going Research (cont.)

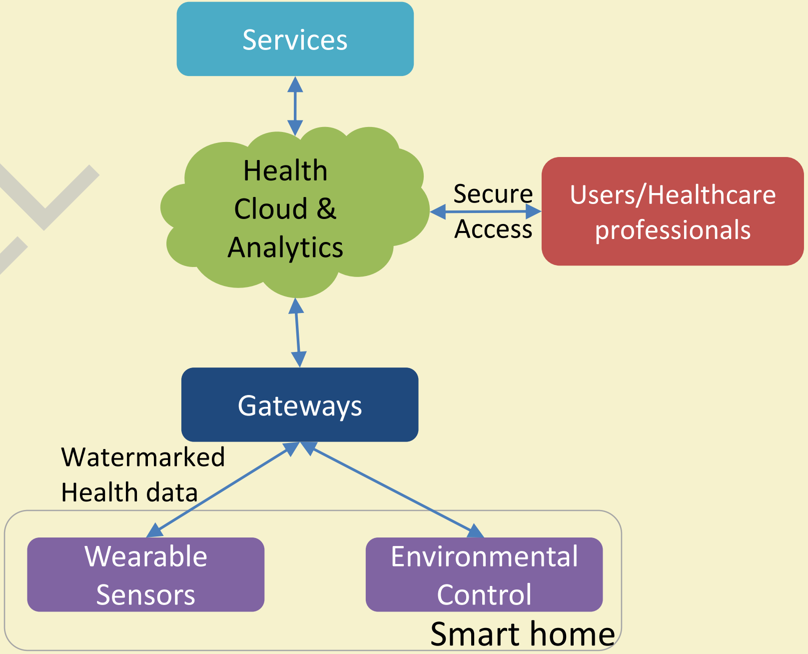
- Data Processing [Karnouskos *et al.*, 2014]
  - <u>Functional group & block</u>: In devices or in cloud
  - <u>Services</u>: Simple filtering to complex analytics
  - <u>*Complex event processing (CEP)*</u>: Real-time correlation & aggregation of event data
  - <u>Rule-based deployment</u> on incoming events
  - <u>API-based facility to create, modify, or delete rules</u>



Source: Karnouskos et al., 2014

IIoT Processing: On-going Research (cont.)

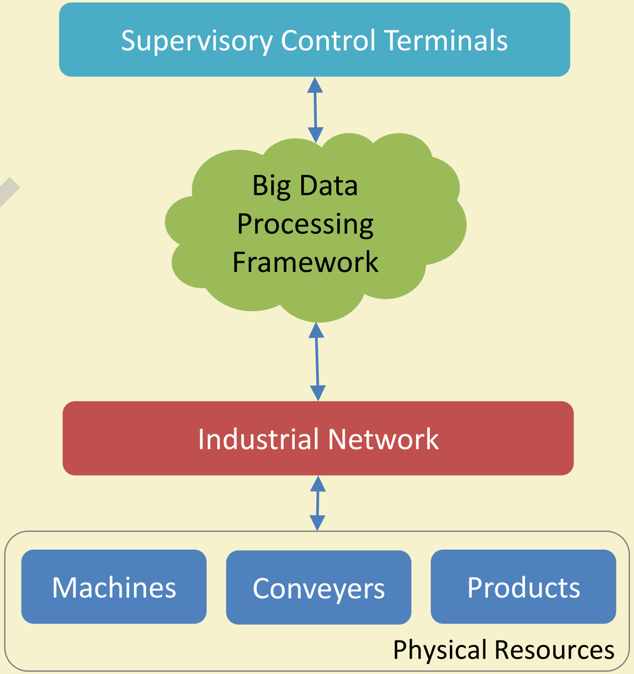
- HealthIIoT [Hossain et al., 2016]
  - Health data collected by sensor-equipped wearable devices
  - Cloud-based analytics for <u>clinical prediction</u>
  - Incorporates *watermarking* & *user identification* in the health data to enhance security
  - Cloud-based dynamic resource management & service provisioning
  - <u>Health condition monitoring</u> by in-loop healthcare professional



Source: Hossain et al., 2016

IIoT Processing: On-going Research (cont.)

- Self-organized Multi-agent System in Smart Factory [Wang et al., 2016]
  - Components: *cloud, industrial network, smart terminals*
  - Increased flexibility due to distributed cooperation and autonomous decision making framework
  - Self-organizing is achieved by intelligent negotiations between agents
  - Cloud-based big data processing framework assists the self-organization & supervisory control



Source: Wang et al., 2016

IIoT Processing: On-going Research (cont.)

- Line Information System Architecture (LISA) [Theorin et al. , 2017]
  - Event-driven information system
  - Loosely-coupled system with prototype-oriented information model
  - Components
    - *LISA events*: machine state change, occurrence of new information
    - *Message bus*: enterprise service bus with standard & structured framework for message routing
    - *Communication end-points*: interoperable communication for services
    - *Service end-points*: interoperable communication to standard interfaces

Source: Theorin et al., 2017

References

[1] A. Dey, K. Stuart and M. E. Tolentino, “Characterizing the impact of topology on IoT stream processing,” *in Proc. of the IEEE World Forum on Internet of Things (WF-IoT)*, 2018, pp. 505-510.

[2] C. E. Kaed, I. Khan, A. Van Den Berg, H. Hossayni and C. Saint-Marcel, “SRE: Semantic Rules Engine for the Industrial Internet-Of-Things Gateways,” in *IEEE Transactions on Industrial Informatics*, vol. 14, no. 2, pp. 715-724, 2018.

[3] A. Akbar, F. Carrez, K. Moessner, J. Sancho and J. Rico, “Context-aware stream processing for distributed IoT applications," *in Proc. of the IEEE World Forum on Internet of Things (WF-IoT)*, 2015, pp. 663-668.

[4] L. Zhou, D. Wu, J. Chen and Z. Dong, “When Computation Hugs Intelligence: Content-Aware Data Processing for Industrial IoT," in *IEEE Internet of Things Journal*, vol. 5, no. 3, pp. 1657-1666, 2018.

[5] H. Wang, O. L. Osen, G. Lit, W. Lit , H.-N. Dai, W. Zeng, “Big Data and Industrial Internet of Things for the Maritime Industry in Northwestern Norway,” in *Proc. IEEE TENCON*, Macao, China, 2015.

[6] A. W. Colombo, S. Karnouskos and T. Bangemann, “Towards the Next Generation of Industrial Cyber-Physical Systems,”*Industrial Cloud-Based Cyber-Physical Systems*, A. W. Colombo et al. (eds.), Springer, 2014.

References

[7] S. Karnouskos, A. W. Colombo, T. Bangemann, K. Manninen, R. Camp, M. Tilly, M. Sikora, F. Jammes, J. Delsing, J. Eliasson, P. Nappey, J. Hu and M. Graf, "The IMC-AESOP Architecture for Cloud-Based Industrial Cyber-Physical Systems,"*Industrial Cloud-Based Cyber-Physical Systems*, A. W. Colombo et al. (eds.), Springer, 2014.

[8] M. S. Hossain and G. Muhammad, "Cloud-assisted Industrial Internet of Things (IIoT) – Enabled framework for health monitoring," *Computer Networks*, vol. 101, pp. 192-202, 2016.

[9] S. Wang, J. Wan, D. Zhang, D. Li, C. Zhang, "Towards smart factory for industry 4.0: a self-organized multi-agent system with big data based feedback and coordination," Computer Networks, vol. 101, pp. 158-168, 2016.

[10] A. Theorin, K. Bengtsson, J. Provost, M. Lieder, C. Johnsson, T. Lundholm, and B. Lennartson, "An event-driven manufacturing information system architecture for Industry 4.0," International Journal of Production Research, vol 55, no. 5, pp. 1297-1311, 2017.

[11] MIDAS: IoT/M2M Platforms, Web: https://www.happiestminds.com/solutions/iot-service-platform-midas/

Thank You!!

FarmBeats

- Data-driven precision agriculture
- *Challenges*: *Intra- & Inter-farm connectivity management, data collection* and *energy management*
- *Components*: Soil sensors, camera, UAVs, weather station, IoT gateway, IoT base station, cloud-services
- Suitable for large-scale long-term deployment
- Gateway incorporates weather-aware decisions & UAV flight planning

Source: Vasisht et al., 2017

FarmBeats (cont.)

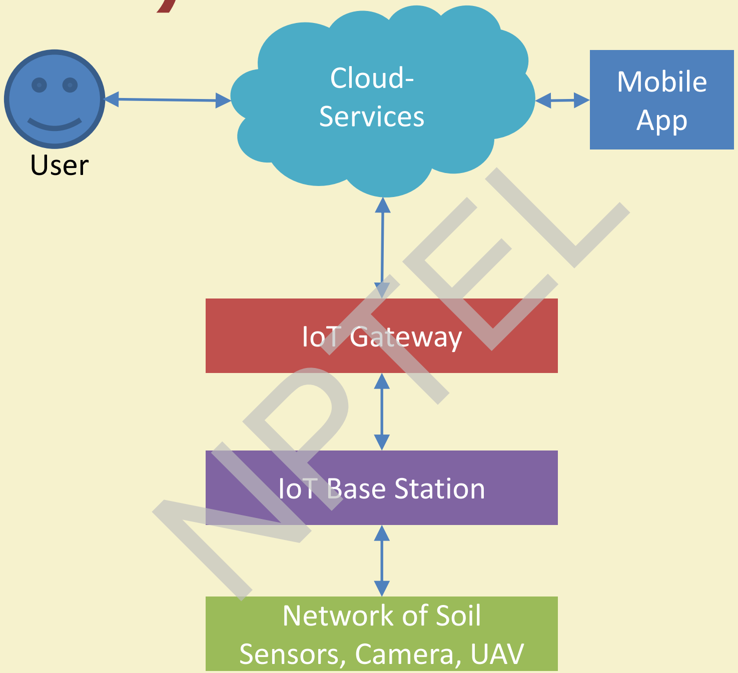


Fig: FarmBeats Architecture

Source: Vasisht et al., 2017

Smart Water Management Platform (SWAMP)

- Irrigation management for different types of crops & climate in different countries
- Services
  - *Entirely replicable services*: interaction with virtual entities, storage, analytics
  - *Fully customizable services*: water management & distribution
  - *Application specific services*: custom requirement specific & supports different architectures

Source: Kamienski et al., 2018

Smart Water Management Platform (SWAMP) (contd.)

- Components: *sensors, virtual entity, analytics & learning, data management, service management*
- SWAMP enables a smart management layer between the *water distribution network & farm-based irrigation system*

Source: Kamienski et al., 2018

SWAMP (cont.)

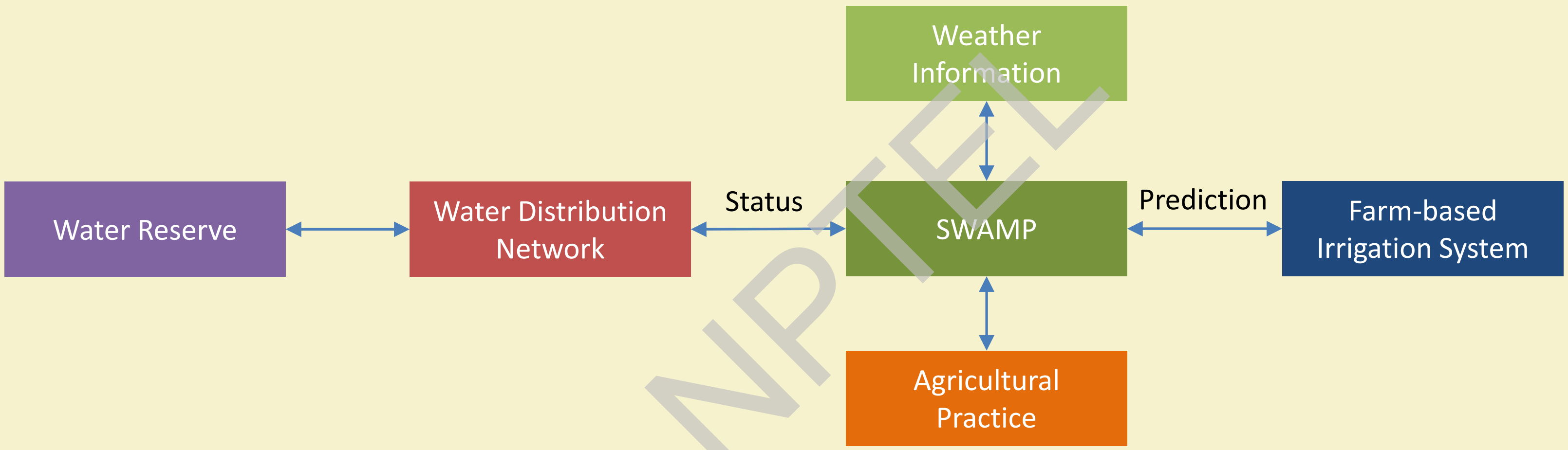


Fig: SWAMP Architecture

Source: Kamienski et al., 2018

AR Drones-based Precision Agriculture

- Precise fertilizer spray to the weeds
- Components: AR Drones, laptop, sprayer installed in a tractor
- The video processing module deployed in the laptop detects the weeds
- The precision sprayer installed in the tractor actuated according to the locations detected by the video processing module

Source: Cambra et al., 2018

AR Drones-based Precision Agriculture (cont.)

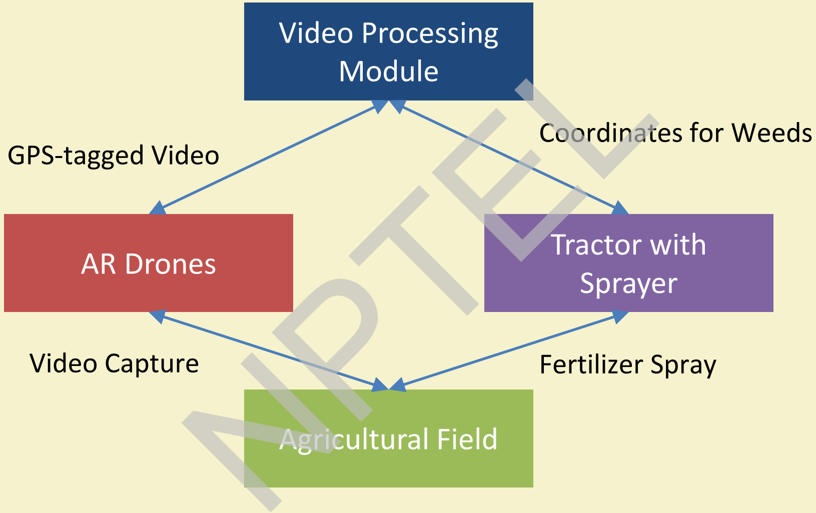


Fig: AR Drone-based Precision Agriculture

Source: Cambra et al., 2018

Vineyard Health Monitoring

- Challenge: Different variety of grape needs different climate conditions
- Real-time sensing and monitoring of vineyards
- Analytics to empower understanding of plant growth according to soil and climatic conditions
- Objective:
  - Increase yield, quality of grapes, with optimal use of water
  - Disease detection & control, optimal use of fertilizers

Source: SensorCloud by LORD MicroStrain

Vineyard Health Monitoring (cont.)

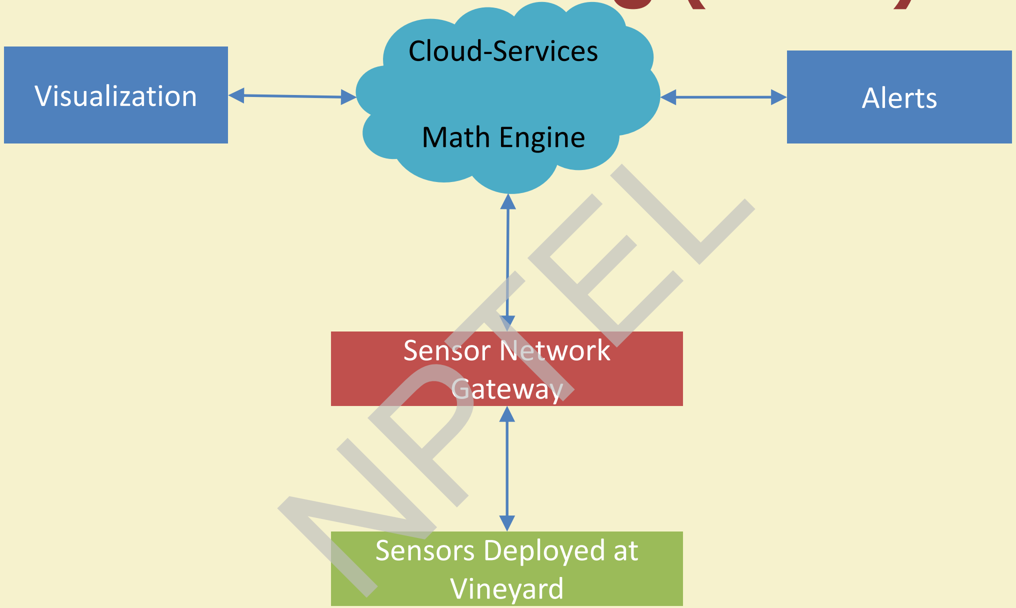


Fig: Vineyard Health Monitoring Framework

Source: SensorCloud by LORD MicroStrain

SmartSantander

- IoT-based smart city deployment platform for large-scale applications
- Design considerations –
  - experimentation realism
  - heterogeneity
  - scale
  - mobility
  - reliability
  - user involvement

Source: SensorCloud by LORD MicroStrain

SmartSantander (contd.)

- Components – IoT nodes, repeaters, and IoT gateways
- Architectural layers: *Authentication, Authorization and Accounting (AAA) subsystem, Testbed management subsystem (MSS), Experimental support subsystem (ESS)*, and *Application support subsystem (ASS)*

Source: SensorCloud by LORD MicroStrain

SmartSantander (cont.)

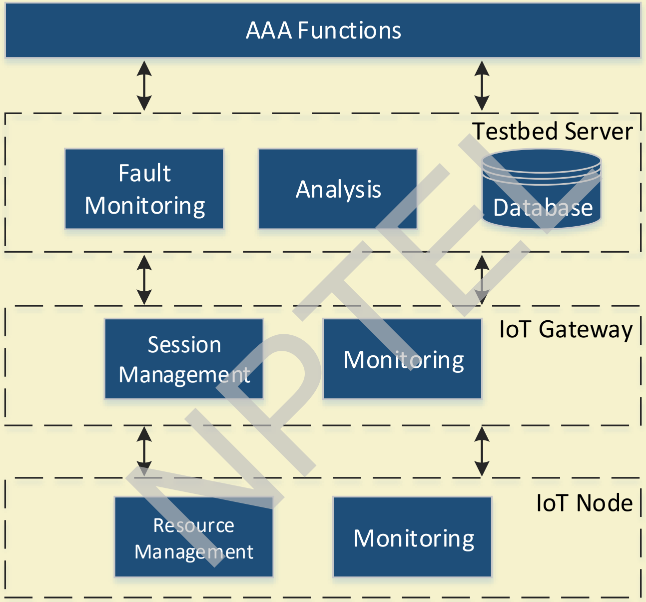


Fig: SmartSantander

Source: SensorCloud by LORD MicroStrain

iRobot-Factory: Cognitive Manufacturing

- Application of cognitive intelligence & edge computing for improved manufacturing
- Automation of the production line by information interaction & data fusion
- Components:
  - *Intelligent terminal*: Tasked with sensing user's emotion & request computing resources accordingly

Source: Hu et al., 2018

iRobot-Factory: Cognitive Manufacturing (contd.)

- *System Management*: Real-time analysis on collected data – emotion data, factory data
- *Edge Computing Node*: Enables low-latency response & decision system at the edge
- *Cognitive Engine*: Cloud-based high performance long-term data analytics using artificial intelligence techniques
- *Intelligent Device Unit*: The hardware assembler and manufacturing unit
- *Production Line Layer*: Production line sequencing with intelligent conveyer units

Source: Hu et al., 2018

iRobot-Factory (cont.)

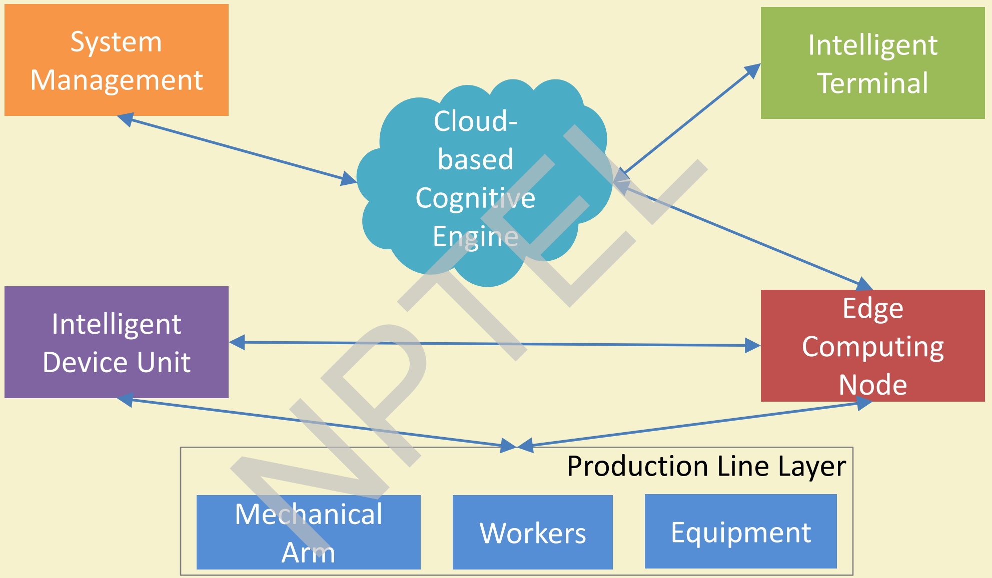


Fig: iRobot-Factory System Architecture

Source: Hu et al., 2018

Big Data Driven Smart Manufacturing

- Challenges: Existing investments, risk & regulation for new technology, lack of skill, mixed workplace
- Different phases of smart manufacturing
  - *Phase 1 - integration of data and contextual information*: gather data from sensors placed at different parts of the industry to have a contextual view
  - *Phase 2 – synthesis & analysis*: processing of data to build knowledge required for decision making
  - *Phase 3 – innovation in process & production*: using knowledge and intelligence to find new insight and use it for future innovation

Source: Donovan et al., 2015

Big Data Driven Smart Manufacturing (cont.)

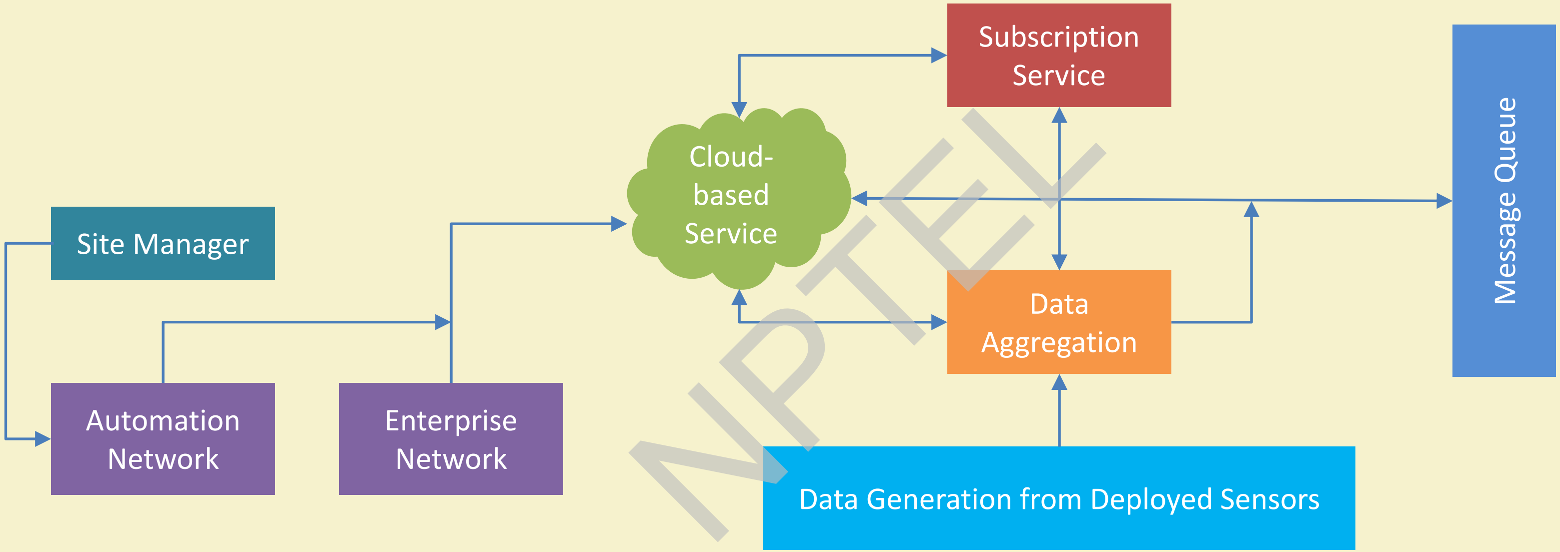


Fig: Architecture for Big Data Processing in Smart Manufacturing

Source: Donovan et al., 2015

Smart Warehousing

- REST-based framework
- Data collection module:
  - Uniquely identifiable objects with RFID tags, sensors
  - Database for storing the information
  - Authenticated & secure access
- Administrative module:
  - Organize & process data, decision making
  - Generating & controlling the events in real-time
  - Dynamic operational parameters & history-based decision making

Source: Jabbar et al., 2016

Smart Warehousing (cont.)

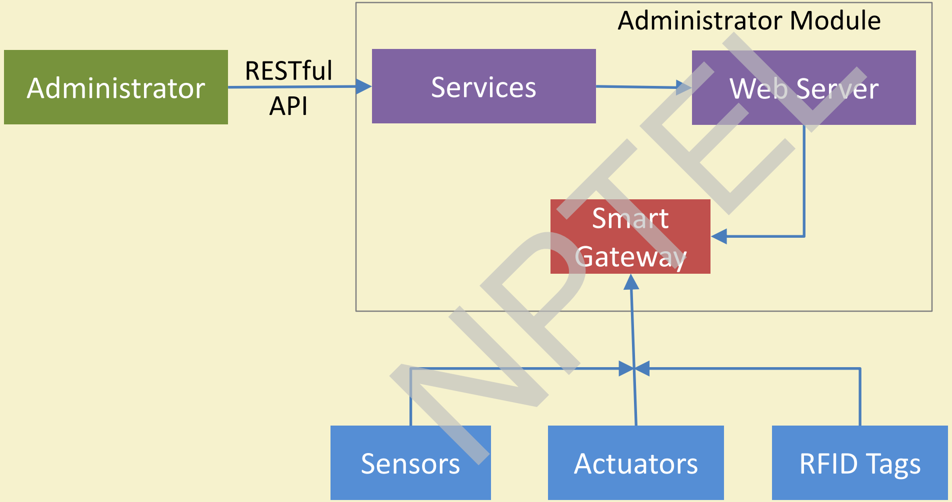


Fig: System Architecture for Smart Warehousing

Source: Jabbar et al., 2016

Industrial Manufacturing

- Cloud computing & IoT services-based
- User entities:
  - *Providers:* service offering organization
  - *Consumers:* service subscribers
  - *Operators:* middle-man, who provisions the services

Source: Tao et al., 2014

Industrial Manufacturing (contd.)

- Workflow:
  - Phase 1: collection of the service offerings & infrastructure
  - Phase 2: virtualization, allocation & management of services
  - Phase 3: on-demand service provisioning
- Layers: (bottom) IoT layer, (middle) Service layer, (top) Application layer, (cross-layer) bottom support layer (knowledge, cloud security, wider internet)

Source: Tao et al., 2014

Industrial Manufacturing (cont.)

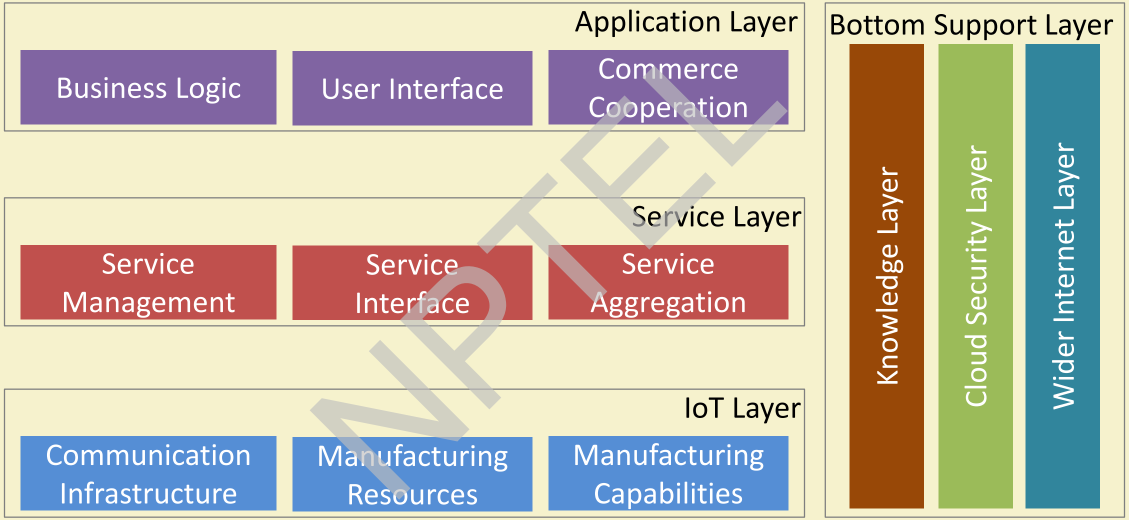


Fig: System Architecture for Industrial Manufacturing

Source: Tao et al., 2014

References

[1] D. Vasisht, Z. Kapetanovic, J. ho Won, X. Jin, R. Chandra, A. Kapoor, S. N. Sinha, M. Sudarshan, and S. Stratman, “FarmBeats: An IoT platform for data-driven agriculture,” in *Proc. of USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, Boston, MA, USA, 2017, pp. 515-529.

[2] C. Kamienski, J.-P. Soininen, M. Taumberger, S. Fernandes, A. Toscano, T. S. Cinotti, R. F. Maia, and A. T. Neto, “SWAMP: an IoT-based smart water management platform for precision irrigation in agriculture,” in *Proc. of Global IoT Summit*, Bilbao, Spain, 2018, pp. 1-6.

[3] C. Cambra, J. R. D´ıaz, and J. Lloret, “Deployment and performance study of an Ad Hoc network protocol for intelligent video sensing in precision agriculture,” in *Proc. of Ad-hoc Networks and Wireless*. Springer Berlin Heidelberg, 2015, pp. 165–175, LNCS 8629.

[4] Case study - vineyard health management with wireless sensor networks and SensorCloud. Web: http://www.sensorcloud.com/static/files/documents/SolutionBrief SCVineyard.pdf

[5] L. Sanchez, L. Muoz, J. A. Galache, P. Sotres, J. R. Santana, V. Gutierrez, R. Ramdhany, A. Gluhak, S. Krco, E. Theodoridis, and D. Pfisterer, “Smartsantander: Iot experimentation over a smart city testbed,” *Computer Networks*, vol. 61, pp. 217-238, 2014.

References (cont.)

[6] L. Hu, Y. Miao, G. Wu, M. M. Hassan, and I. Humar, “iRobot-Factory: An intelligent robot factory based on cognitive manufacturing and edge computing,” *Future Generation Computer Systems*, 2018.

[7] P. O’Donovan, K. Leahy, K. Bruton and D. T. J. O’Sullivan, “An industrial big data pipeline for data-driven analytics maintenance applications in large-scale smart manufacturing facilities,” *Journal of Big Data*, vol. 2, no. 25, 2015.

[8] G. Han, A Qian, J. Jiang, N. Sun, L. Liu, “A grid-based joint routing and charging algorithm for industrial wireless rechargeable sensor networks,” Computer Networks, vol. 101, pp. 19-28, 2016.

[9] S. Jabbar, M. Khan, B. N. Silva, K. Han, "A REST-based industrial web of things’ framework for smart warehousing," The Journal of Supercomputing, 2016 [DOI: 10.1007/s11227-016-1937-y]

[10] F. Tao, Y. Cheng, L. D. Xu, L. Zhang, B. H. Li, “CCIoT-CMfg: Cloud Computing and Internet of Things-Based Cloud Manufacturing Service System,” IEEE Transactions on Industrial Informatics, vol. 10, no. 2, pp. 1435-1442, 2014.

Thank You!!

IIT KHARAGPUR

NPTEL ONLINE CERTIFICATION COURSES

NPTEL

IIoT Process Control

Dr. Sudip Misra

Professor

Department of Computer Science and Engineering

Indian Institute of Technology Kharagpur

Email: smisra@sit.iitkgp.ernet.in

Website: http://cse.iitkgp.ac.in/~smisra/

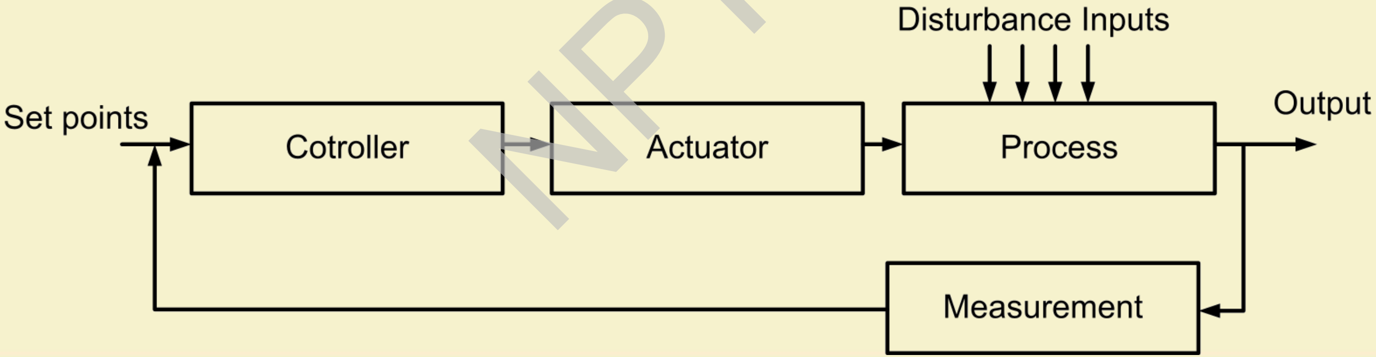
Research Lab: cse.iitkgp.ac.in/~smisra/swan/

What are Industrial Control Systems?

- Different types of electro-mechanical instruments and the associated systems used in industries to control various industrial units or *processes*
- Comprise of four major components:
  - Process Variables - Values of process parameters measured using devices such as sensors
  - Set Points - Standard values of the process parameters for controlled operation of the process

What are Industrial Control Systems? (Contd.)

- Controllers – For taking action decisions based on comparison of process variables with set points
- Manipulating Variables – Process variables modified based on controller decisions to manipulate the process



Control Loops

- Fundamental element of industrial control systems for automatic control of industrial process variables
- Two types:
  - Open Loop Control – Control decision independent of process variable
  - Closed Loop / Feedback Control – Control decision depends on the measured value of process variable

Types of Industrial Control Systems

- Programmable Logic Controllers (PLCs)
- Distributed Control Systems (DCS)
- Supervisory control and Data Acquisition (SCADA)

Programmable Logic Controllers (PLCs)

- An industrial control system based on programming logic capable of –
  - Monitoring the industrial processes
  - Taking control actions based on some predefined computer program
- Comprises of a processor unit, memory unit, power supply and communication modules
- Used in assembly lines and robotic manufacturing devices

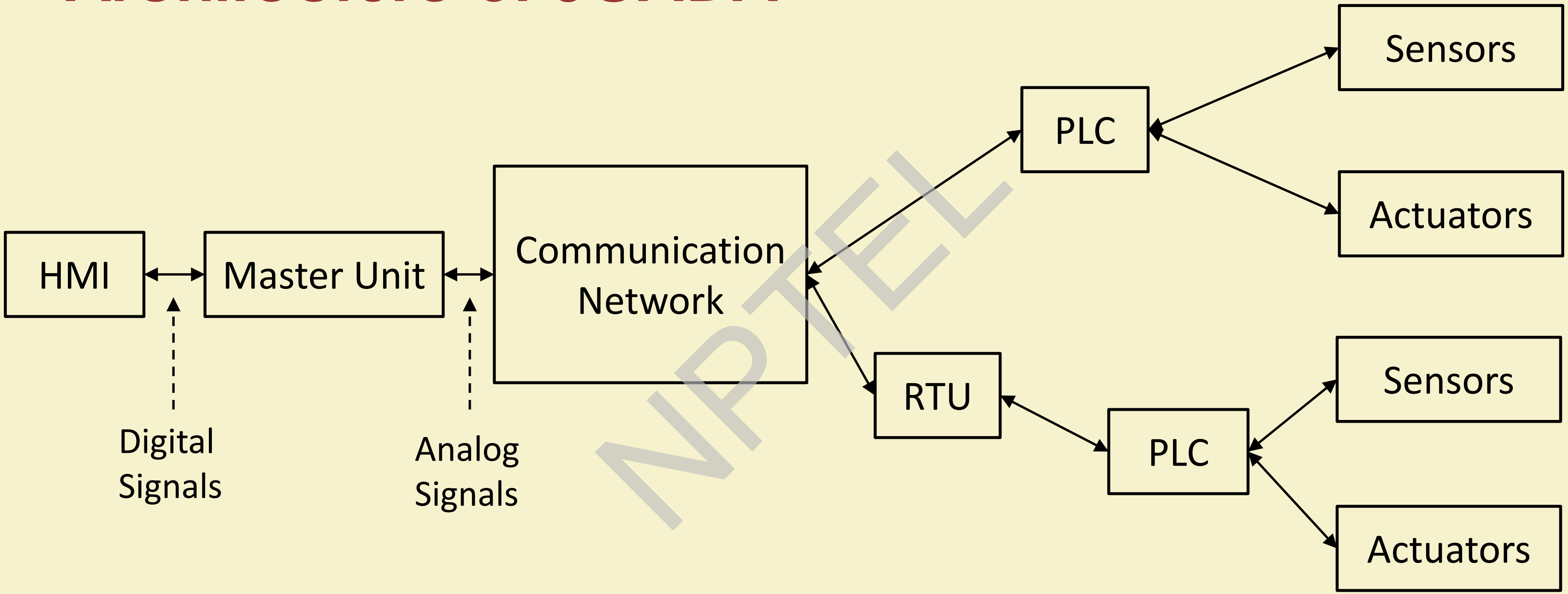
Distributed Control Systems (DCS)

- Specially designed control systems used to control highly distributed plants having huge number of control loops
- Improved reliability due to distributed control
- Main components are –
  - Central supervisory controller
  - Distributed controllers
  - Field devices such as sensors and actuators
  - High-speed communication network

Supervisory control and Data Acquisition (SCADA)

- Industrial process automation system used in automatic traffic management, water distribution, electric power grids, etc
- Main components are:
  - Sensors and Control Relays
  - Remote Telemetry Units (RTUs)
  - SCADA master units
  - Human-Machine Interface (HMI)
  - Communication Infrastructures

Architecture of SCADA



References

[1] Groover, M. P. (2007). *Automation, production systems, and computer-integrated manufacturing*. Prentice Hall Press.

[2] Bolton, W. (2015). *Programmable logic controllers*. Newnes.

[3] D'Andrea, Raffaello (9 September 2003). "Distributed Control Design for Spatially Interconnected Systems". *IEEE Transactions on Automatic Control*.

[4] Boyer, S. A. (2009). *SCADA: supervisory control and data acquisition*. International Society of Automation.

[5] Alur, R., Arzen, K. E., Baillieul, J., & Henzinger, T. A. (2007). *Handbook of networked and embedded control systems*. Springer Science & Business Media.

Thank You!!