IoT Workshop

Trygve Laugstøl <trygvis@trygvis.io>

What is IoT

What is IoT

▶ Not "a computer connected to the internet"
  ▶ Then it is really just another computer connected to the internet
▶ Must be something else
  ▶ It is simply devices that are resource constrained
    ▶ Usually in more than one way
▶ Autonomous operation, the connection might not be permanent

IoT is just a concept

▶ *The Internet of Things (IoT) is the network of physical devices, vehicles, home appliances and other items embedded with electronics, software, sensors, actuators, and connectivity which enables these objects to connect and exchange data.*[1]

---

[1] Wikipedia "Internet of Things"

What is an IoT Device?

What is an IoT Device?

▶ Constrained in (one or more of):
  ▶ Memory
  ▶ CPU
  ▶ Network bandwidth and/or latency
  ▶ Storage
▶ Has connectivity
  ▶ Bluetooth
  ▶ Wi-Fi
  ▶ NB-IoT
  ▶ LTE Cat-M
  ▶ LoRA
  ▶ Proprietary radio

IoT Devices - Bluetooth 4/5 chips

Chip	CPU	Freq	RAM	Flash	Price
nRF52810	Cortex-M4	64 MHz	24k	192k	$1.88
nRF52832	Cortex-M4F		32k	256k	$2.54
			64k	512k	$2.59
nRF52840	Cortex-M4F		256k	1024k	$3.85

▶ nRF52810: High performance, entry-level Bluetooth 4/ANT/2.4GHz SoC
▶ nRF52832: High performance Bluetooth 4/ANT/2.4GHz SoC
▶ nRF52840: Advanced multi-protocol System-on-Chip Supporting: Bluetooth 5, ANT/ANT+, 802.15.4 and 2.4GHz proprietary

IoT Devices - LoRA

Modules

Module	Data Rate	Price
RN2483A-I/RM104		$12.05 @ 250
CMWX1ZZABZ-078	SX1276	$10.74 @ 1000
RF-LORA-868-SO	SX1272	$16.55 @ 1000

Chips

Chip	Price
SX1281	$3.23
SX1272	$4.25
SX1276	$4.25
SX1279	$4.74

IoT Devices - NB-IoT

Module	Price
uBlox SARA-N210	~$10 @ 100
Sierra Wireless HL7800_1103933	$15.72

IoT Devices - Wi-Fi

Chip	CPU	Freq	ROM	RAM	Price
ESP8266	Tensilica L106	160 MHz	N/A	~50 kB	< $1

ESP32 - dual cpu, Wi-Fi, Bluetooth 4 ESP32-D0WDQ6 2x Xtensa @ 160MHz $ 4.53 @ 10

ESP8266 details - Power usage

State	Current usage
Off	0.5 μA
Deep sleep with RTC	20 μA
Light sleep (with Wi-Fi)	1 mA
Sleep with peripherials	15 mA
TX	170 mA

Going back to basics

What is the internet again?

OSI model

1. Physical Layer
2. Data Link Layer
3. Network Layer
4. Transport Layer
5. Session Layer
6. Presentation Layer
7. Application Layer

▶ Wikipedia: OSI model
▶ Wikipedia: OSI model#Examples

Layer 1: Physical Layer

▶ 10BASE5, 10BASE2
▶ 10BASE-T / 100BASE-TX / 1000BASE-TX
▶ 802.11a/b/g/n PHY
▶ RS-232

Layer 2: Data Link Layer

- Ethernet
- WiFi
- Bluetooth
- Token Ring

Layer 3: Network Layer

- IP
- ICMP
- IPX

Layer 4: Transport Layer

- TCP
- UDP

Layer 5: Session Layer

- "sockets"
- NetBIOS

Layer 6: Presentation Layer

- ▶ SSL

Layer 7: Application Layer

- HTTP
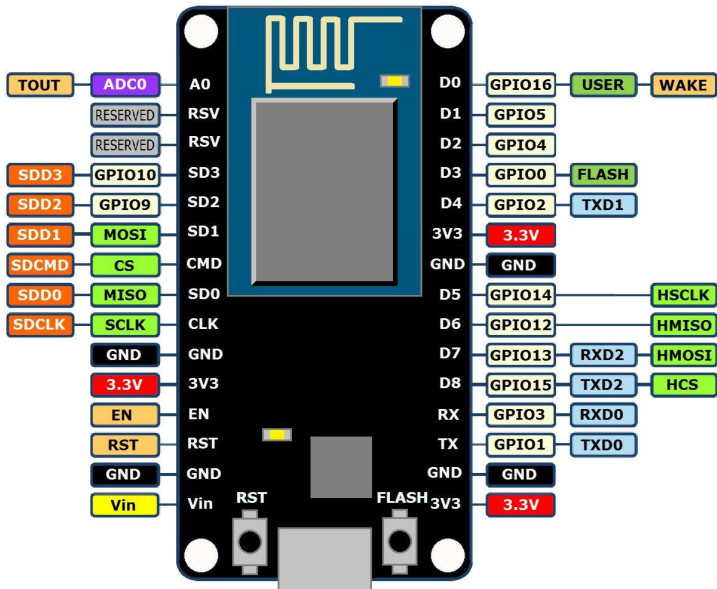- DNS
- MQTT
- CoAP
- (everything else..)

Details: IP

bit	0		7	8		15	16			31
0	version	len		TOS				full length of packet		
4		identification					X¸D¸M		fragment Offset	
8	time to live (TTL)			protocol				header checksum		
12	source IP address									
16	destination IP address									
20	IP options (variable length)									
	payload									

Details: UDP

Offsets	Octet	0								1								2								3							
Octet	Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0	0	source port																destination port															
4	32	length																checksum															

Lecture: ESP8266

NodeMCU hardware

TOUT	ADC0	A0	D0	GPIO16	USER	WAKE
	RESERVED	RSV	D1	GPIO5		
	RESERVED	RSV	D2	GPIO4		
SDD3	GPIO10	SD3	D3	GPIO0	FLASH	
SDD2	GPIO9	SD2	D4	GPIO2	TXD1	
SDD1	MOSI	SD1	3V3	3.3V		
SDCMD	CS	CMD	GND	GND		
SDD0	MISO	SD0	D5	GPIO14		HSCLK
SDCLK	SCLK	CLK	D6	GPIO12		HMISO
	GND	GND	D7	GPIO13	RXD2	HMOSI
	3.3V	3V3	D8	GPIO15	TXD2	HCS
	EN	EN	RX	GPIO3	RXD0	
	RST	RST	TX	GPIO1	TXD0	
	GND	GND	GND	GND		
	Vin	Vin	3V3	3.3V		

ESP-12

ESP8266

UART

CP201x

USB

QSPI

Flash

NodeMCU

ESP8266 software layers

Generic Arduino	Ethernet	ESP APIs
ESP interface		
ESP SDK		GCC libc

Arduino { Generic Arduino, Ethernet, ESP APIs, ESP interface }

ESP8266 Hardware
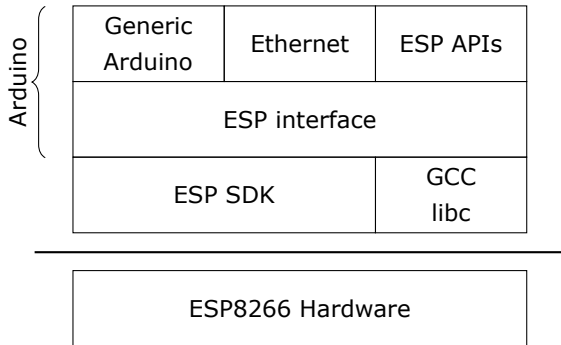
ESP8266 + Arduino

- Standard Arduino IDE
- ESP8266 Arduino core
  - https://github.com/esp8266/Arduino

Arduino IDE


```
Fil Rediger Skisse Verktøy Hjelp

sketch_apr25a

1 void setup() {
2   // put your setup code here, to run once:
3
4 }
5
6 void loop() {
7   // put your main code here, to run repeatedly:
8
9 }
```

Module), 80 MHz, 4M (1M SPIFFS), v2 Higher Bandwidth, Disabled, None, Only Sketch, 921600 on /dev/ttyUSB0
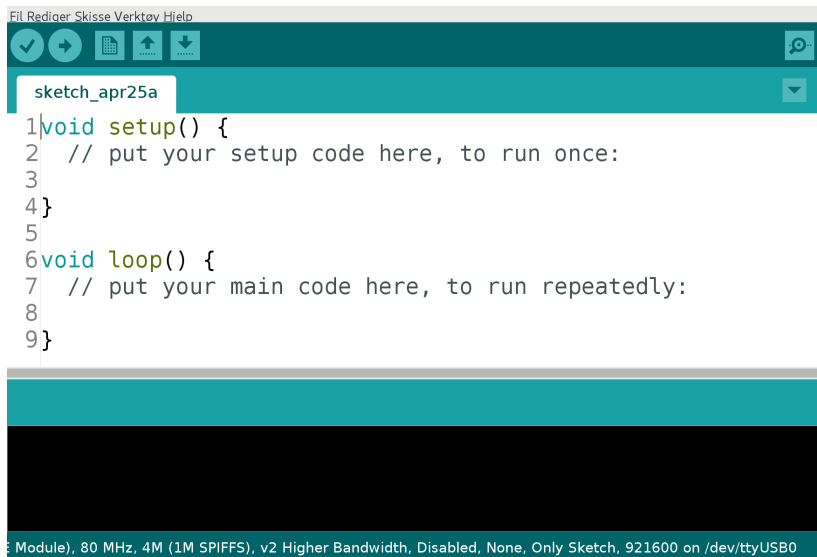
Arduino code structure

```
void setup() {
    // Called once
}

void loop() {
    // Called repeatedly
}
```

Arduino file structure

```
foo/
  foo.ino
  config.h
```

Generic Arduino APIs

```
// Pin: D0, D1, etc.
// Mode: OUTPUT, INPUT, INPUT_PULLUP
void pinMode(uint8_t pin, uint8_t mode);

// State: HIGH, LOW, true/false, 1/0
void digitalWrite(uint8_t pin, uint8_t state);
int digitalRead(uint8_t pin);

unsigned long now millis();
unsigned long now micros();
```

ESP Arduino APIs

```cpp
class {
    void restart();
    uint32_t getFreeHeap();
    uint32_t getChipId();


    ...
} ESP;

// Usage
ESP.restart();
```

ESP Arduino APIs

```
class {
    String macAddress();

    wl_status_t status();
    int32_t RSSI();

    IPAddress localIP();
    IPAddress subnetMask();
    IPAddress gatewayIP();
    IPAddress dnsIP(uint8_t dns_no = 0);


    ...
} WiFi;

// Usage:

Serial.println(WiFi.localIP().toString());
```

Lecture: MQTT

MQTT

▶ *Message Queuing Telemetry Transport*
▶ Wikipedia: MQTT

Device and application architecture with MQTT
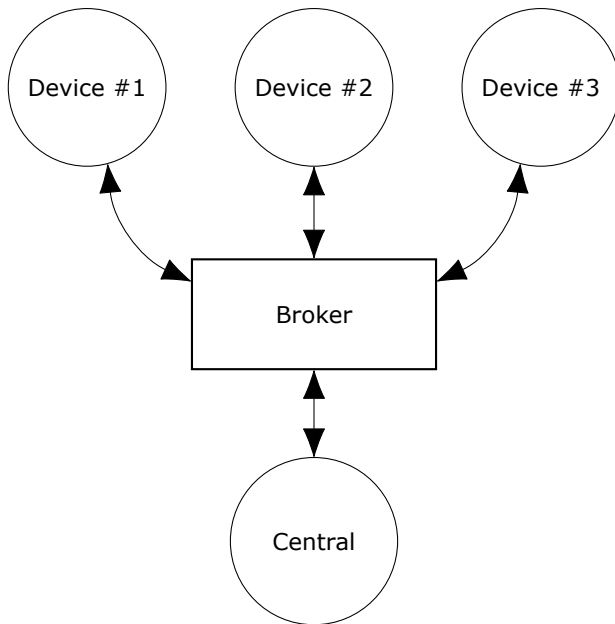
MQTT - Implementations

- Mosquitto
- Eclipse Paho
- RabbitMQ
- ActiveMQ

MQTT Cloud Connectors

- ▶ Cloud
    - ▶ Amazon IoT
    - ▶ Google Cloud IoT
    - ▶ Microsoft Azure IoT
    - ▶ CloudMQTT (at Heroku)
- ▶ DIY
    - ▶ ThingMQ
    - ▶ HiveMQ

MQTT - The protocol

Agents have one of two roles:

- *Client*
  - Publishes *messages*
  - Subscribes / unsubscribes to *topics*
- *Broker* (aka Server)
  - Handles network connections
  - Keeps subscriptions
  - Manages client
    - Disconnects
    - *(last) will*
  - Persistence of retained messages

▶ Size oriented
▶ Flags indicate type of remaining bytes
   ▶ Packet type
   ▶ Topic name
   ▶ Payload

MQTT Connect

- ▶ CONNECT
  - ▶ clientId
  - ▶ username
  - ▶ password
  - ▶ keepAlive

- ▶ Keep alive
  - ▶ PINGREQ
  - ▶ PINGRESP

- ▶ Topic name: `foo/bar/baz`
- ▶ Topic filter
  - ▶ `foo/bar/?`
  - ▶ `foo/#`

Message is kept by the server even after disconnect

▶ CONNECT
▶ PUBLISH
    ▶ RETAIN
    ▶ $app/$device/temperature
    ▶ 22.3
▶ DISCONNECT

Later on:

▶ SUBSCRIBE
    ▶ $app/#/temperature
▶ PUBLISH
    ▶ $app/$device/temperature
    ▶ 22.3

MQTT - The protocol - Will message

Message sent when you disconnect

Client #1:

1. CONNECT
   - ▶ WILL TOPIC: $app/$device/online
   - ▶ WILL PAYLOAD: 0
2. PUBLISH
   - ▶ $app/$device/online
   - ▶ 1
3. DISCONNECT

Broker

1. *To all subscribers* PUBLISH
   - ▶ $app/$device/online
   - ▶ 0

MQTT Topic

The temperature sensor:

- ▶ Publishes on:
  - ▶ `myapp/$device-id/temperature`
  - ▶ `myapp/$device-id/humidity`
  - ▶ `myapp/$device-id/altert`
- ▶ Subscribes to:
  - ▶ `myapp/$device-id/command`

The central application:

- ▶ Subscribes to:
  - ▶ `myapp/#/temperature`
  - ▶ `myapp/#/humidity`
- ▶ Publishes on:
  - ▶ `myapp/$device-id/command`

MQTT on Arduino

PubSubClient is our MQTT client implementation.

```
WiFiClient wifiClient;
PubSubClient mqtt(wifiClient);

void callback(char* topic,
              byte* payload,
              unsigned int length);

void setup() {
    // Configure WiFi
    mqtt.setServer(mqtt_server, 1883);
    mqtt.setCallback(callback);
}
```

MQTT on Arduino

```
void loop() {
    if (!mqtt.connected())
        reconnect();
    else
        mqtt.loop();
    // Do work
}

void reconnect() {
    while (!mqtt.connect(client_id));

    mqtt.subscribe(topic_pattern);
}
```
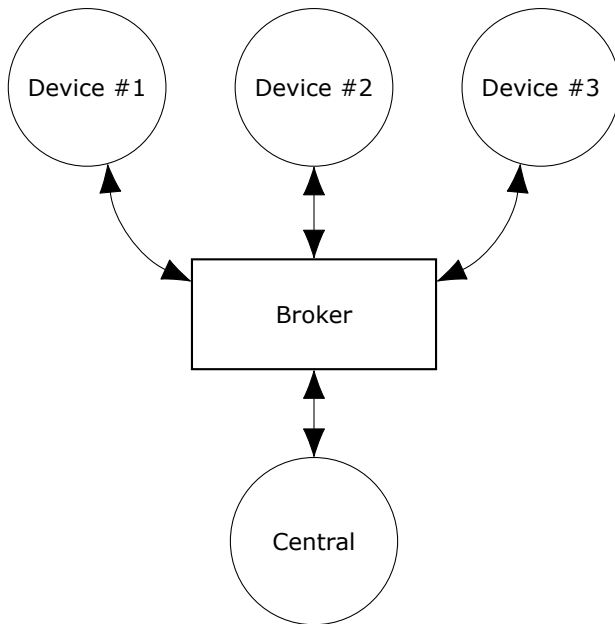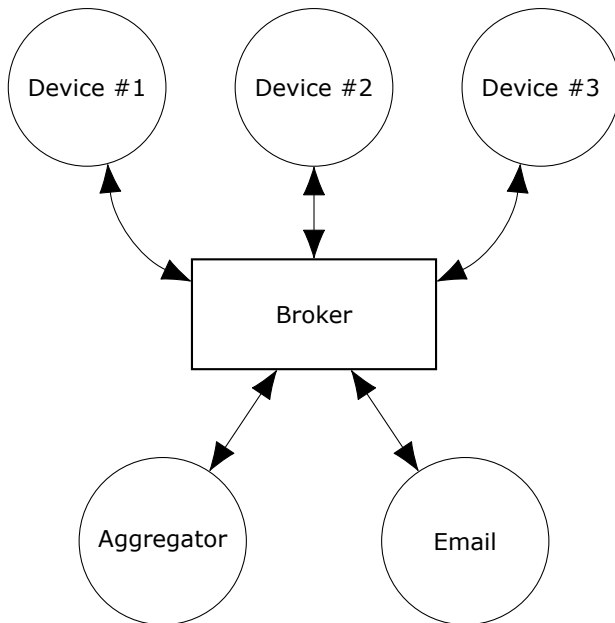
Assignment

- mqtt

MQTT topic architecture

The central application is split:

- ► An aggregating agent:
  - ► `myapp/#/temperature`
  - ► `myapp/#/humidity`
- ► Emailing agent
  - ► `myapp/$device-id/altert`
- ► Publishes on:
  - ► `myapp/$device-id/command`

MQTT topic architecture

MQTT topic architecture

MQTT - Patterns

▶ Combining MQTT and HTTP
▶ Using web sockets transport

Assignment

- `mqtt2`

Assignment

- `mqtt3`