Internet of Things

- ✓ An hands-on, networking oriented perspective

Sesiones

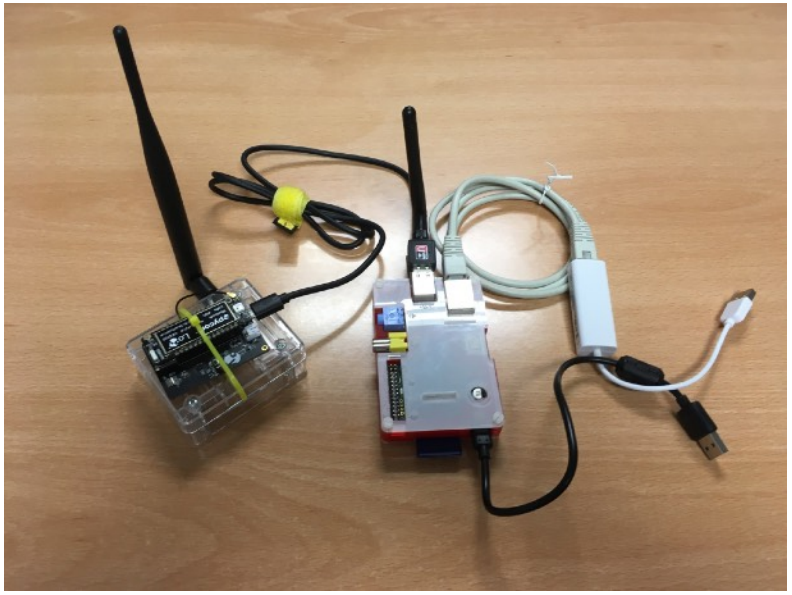
1. Intro a IoT (hoy)
2. MQTT… con ejemplos

MQTTLens

Ofrecido por: MQTTLens

https://chrome.google.com/webstore/detail/mqttlens/

3. Dispositivos y plataformas
4. LoRaWAN



Bibliografia

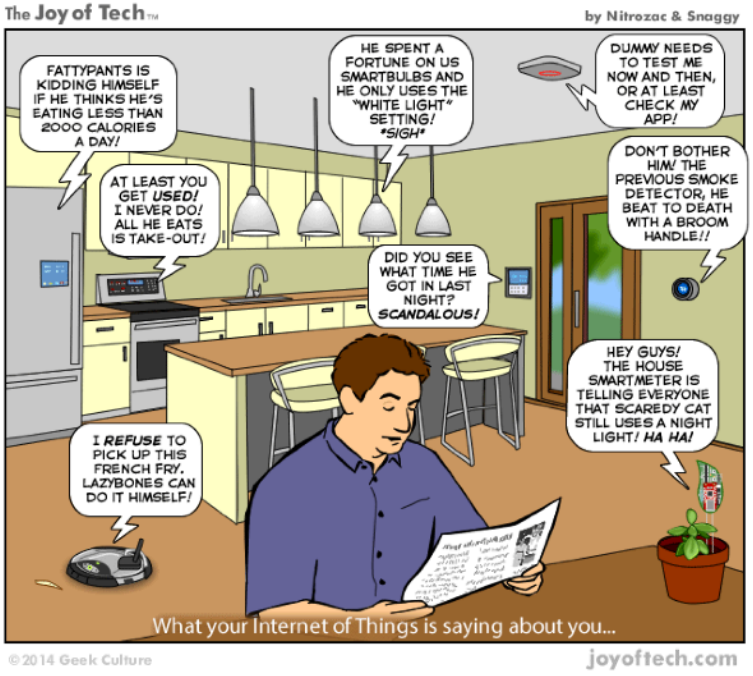
Internet of Things (IoT)

- "The IoT can be viewed as a global infrastructure for the information society, enabling advanced services by interconnecting (physical and virtual) things based on existing and evolving interoperable information and communication technologies (ICT)." Source: Recommendation ITU-T Y.2060

A quick and "physical" definition:
"A network of items—each embedded with sensors—which are connected to the Internet."

- **Cisco**: The Internet of Things (IoT) is the network of physical objects accessed through the Internet, as defined by technology analysts and visionaries. These objects contain embedded technology to interact with internal states or the external environment. In other words, when objects can sense and communicate, it changes how and where decisions are made, and who makes them.

[http://www.cisco.com/web/solutions/trends/iot/overview.html - 21-Jun-2014]

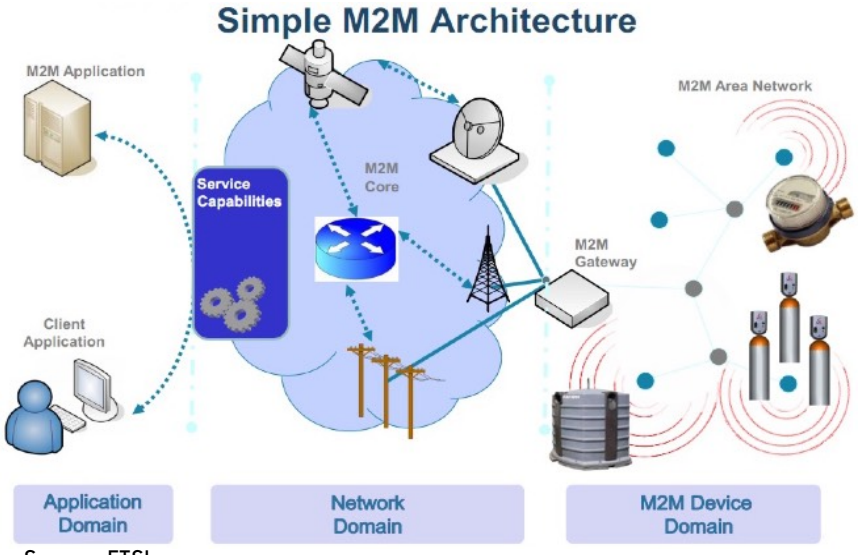


Machine-to-Machine (M2M)

- The Machine-to-Machine Architecture model proposed by ETSI (http://www.etsi.org/) is considered a predecessor of IoT
- Sometimes they are used interchangeably; but M2M is meant for automated interactions between devices while the IoT is a more general term for describing technologies that allow real world data collection, communication, processing and interactions anywhere, anytime and between Anything (Machines, Devices and Human).

M2M Solutions Overview
https://www.youtube.com/watch?v=IwE5YnD5t_s

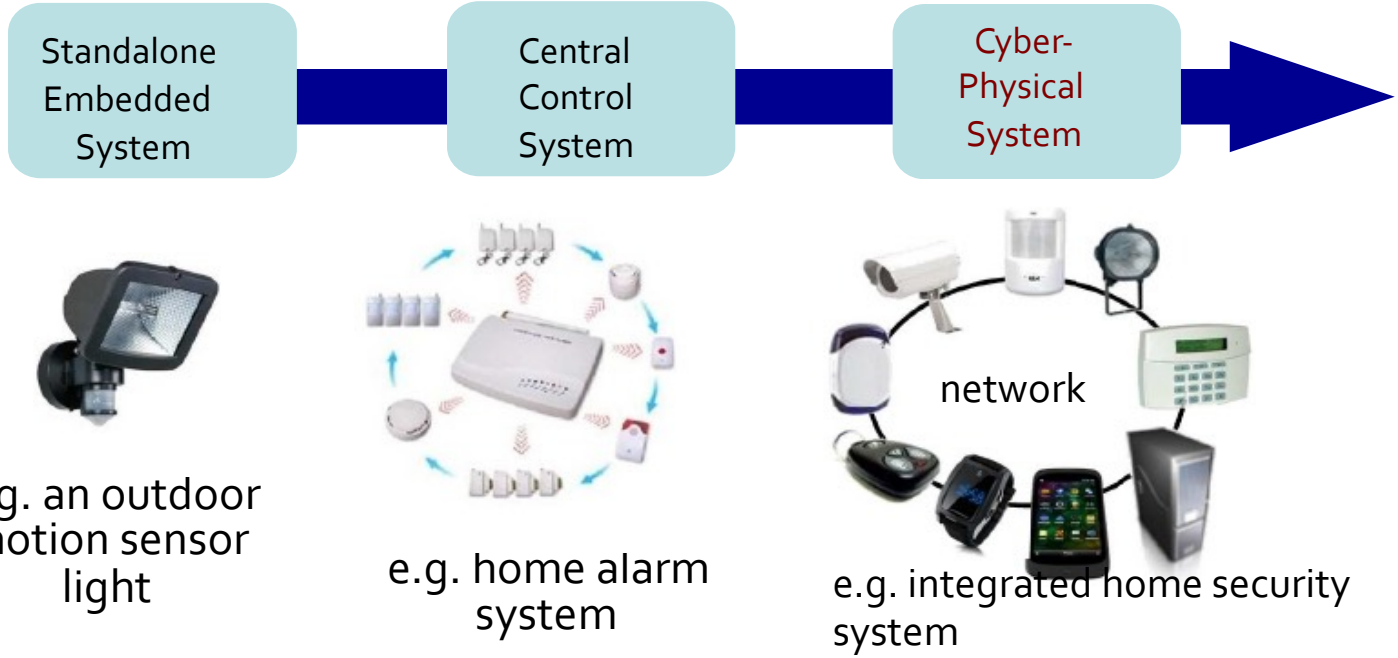
Smart metering
https://www.youtube.com/watch?v=758gVc4AKtY



Source: ETSI

Cyber-Physical Systems

- CPS is commonly used (and preferred over IoT) by the engineering communities (e.g., mechanical engineering, aerospace engineering, aeronautics). It is also used extensively by computer scientists working on embedded systems
- CPS is typically used in the case of systems/problems that involve large scale real-time control (e.g., time critical problems), notably problems that combine control of combined organizational and physical processes.

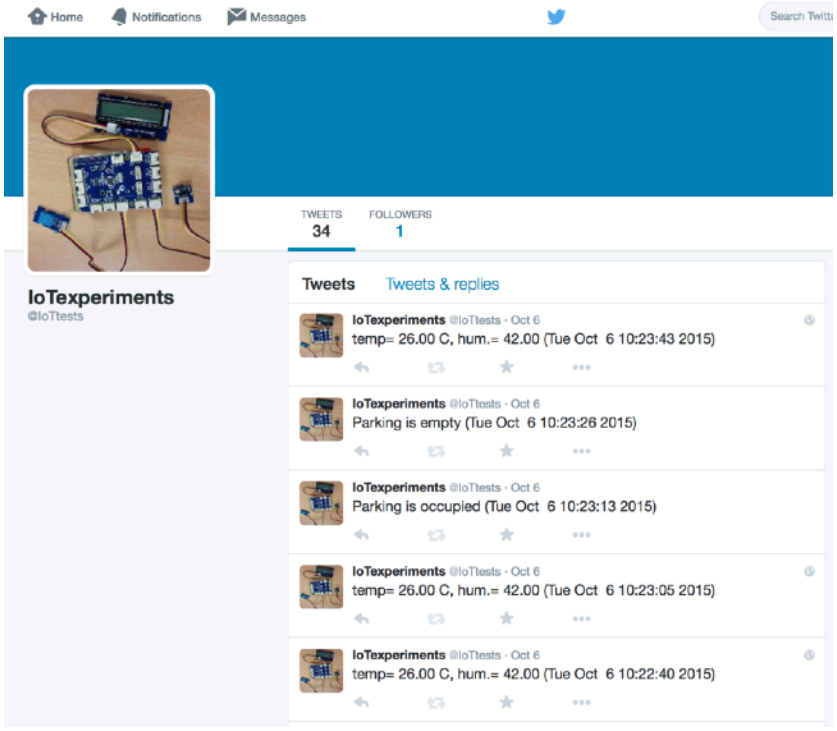


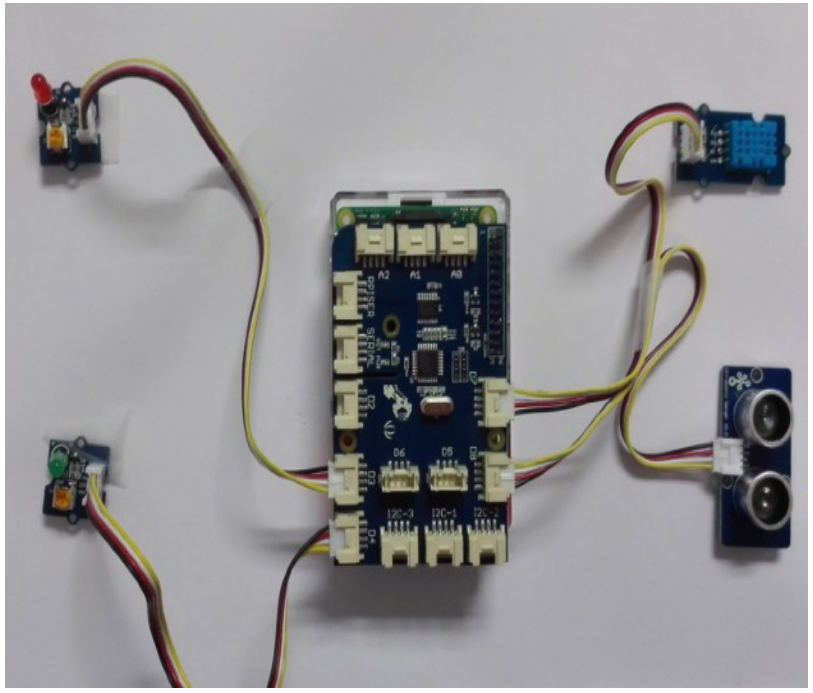
IoT Fundamental characteristics

- **Interconnectivity**: With regard to the IoT, anything can be interconnected with the global information and communication infrastructure.
- **Heterogeneity**: The devices in the IoT are heterogeneous as based on different hardware platforms and networks. They can interact with other devices or service platforms through different networks.
- **Dynamic changes**: The state of devices change dynamically, e.g., sleeping and waking up, connected and/or disconnected as well as the context of devices including location and speed. Moreover, the number of devices can change dynamically.
- **Enormous scale**: The number of devices that need to be managed and that communicate with each other will be at least an order of magnitude larger than the devices connected to the current Internet. The ratio of communication triggered by devices as compared to communication triggered by humans will noticeably shift towards device- triggered communication.

Source: Recommendation ITU-T Y.2060

Let's start with an example…





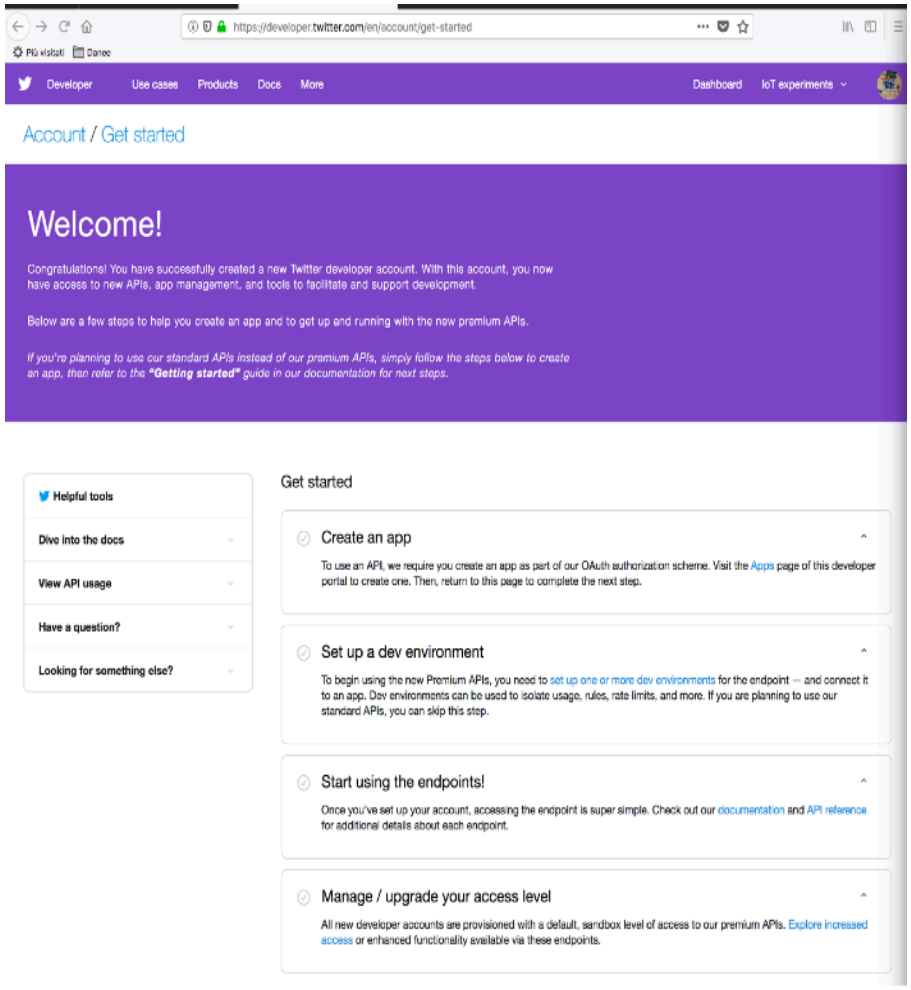
https://www.youtube.com/watch?v=BML-1xEZ9FU

The code...

```
...

# Connect to Twitter
try:
    tapi = twitter.Api(#
    consumer_key='.........................',#
    consumer_secret='.........................',#
    access_token_key='.........................',#
    access_token_secret='.........................')
    print "Twitter Connected."
except:
    print "Unexpected error connecting to Twitter:", sys.exc_info()[0]

time.sleep(1)
print "Starting."
mcycle = 0
occupied = False

while True:
    try:
        [temp,hum] = dht(th_port,0)          # Get the temperature and Humidity from the DHT sensor
        ultra = ultrasonicRead(usonic_port) # Get the distance from the ultrasonic sensor

        print ("(%5.0f) temp= %5.2f C, hum.= %5.2f" % (mcycle, temp, hum))
        mcycle += 1
        if  mcycle >= 20:
            mcycle = 0
            print "Time has come! Let's tweet the temperature and the humidity from the DHT sensor."
            tapi.PostUpdate("temp= %5.2f C, hum.= %5.2f (%s)" % (temp, hum, time.asctime()))

# informing about the parking status...
        if ultra < 10.0:
            print "Parking is occupied"
            if occupied == False:
                occupied = True
                digitalWrite(greenled_port,1)
                digitalWrite(redled_port,0)
                print "Status has changed! Let's tweet it."
                tapi.PostUpdate('{message} ({id})'.format(message="Parking is occupied", id=time.asctime()))
        else:
            print "Parking is empty"
            if occupied == True:
                occupied = False
                digitalWrite(greenled_port,0)
                digitalWrite(redled_port,1)
                print "Status has changed! Let's tweet it."
                tapi.PostUpdate('{message} ({id})'.format(message="Parking is empty", id=time.asctime()))

    except KeyboardInterrupt:
        print "Exiting"
        exit()
    except:
        print "Unexpected error:", sys.exc_info()[0]

    time.sleep(0.5)
```

pip install python-twitter

Creating an app 1/2



App details

The following app details will be visible to app users and are required to generate the API keys needed to authenticate Twitter developer products.

App name (required)

MyOwnDataPublisher

Maximum characters: **32**

Application description (required)

Share a description of your app. This description will be visible to users so this is a good place to tell them what your app does.

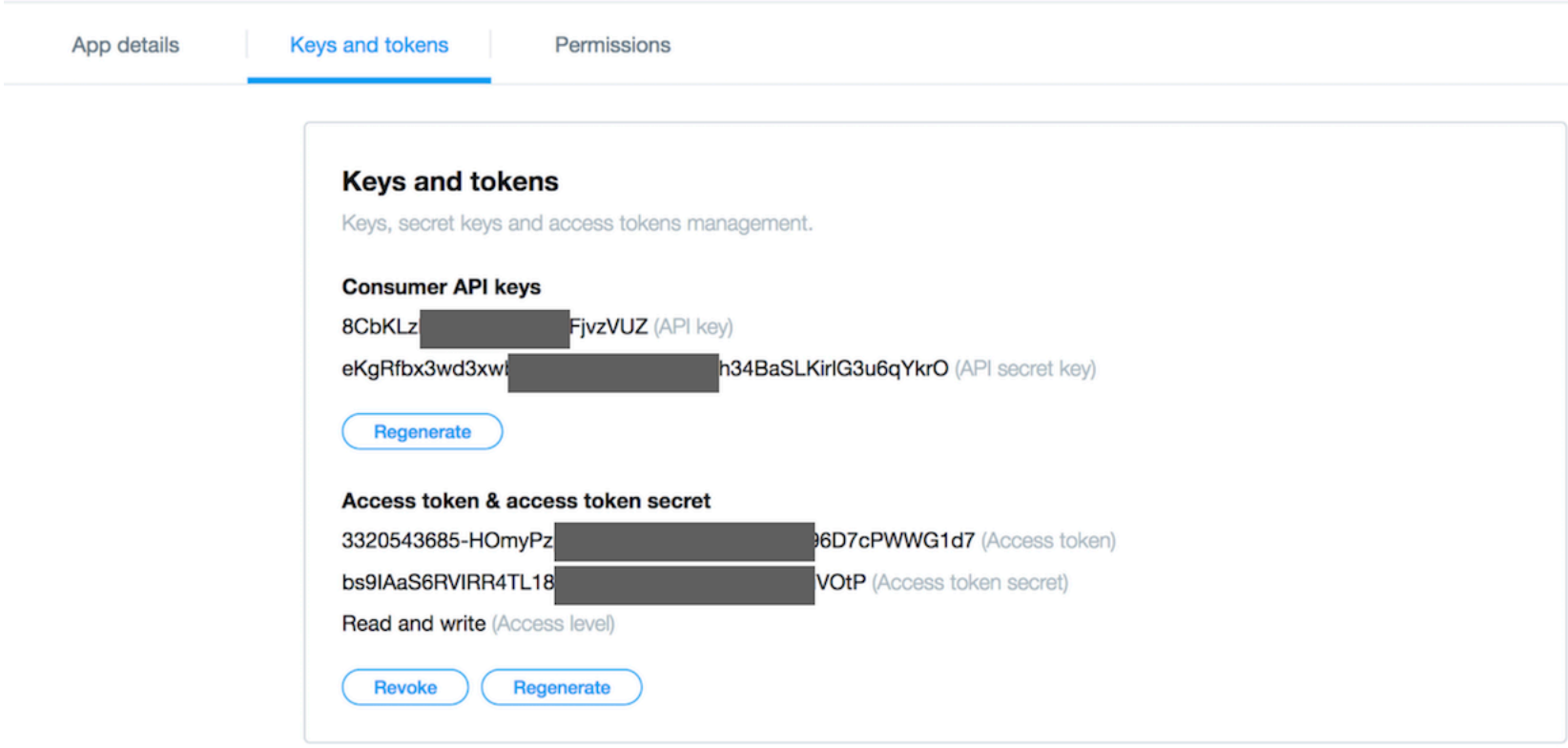
Please be detailed.

Between 10 and 200 characters

Website URL (required)

http://www.grc.upv.es

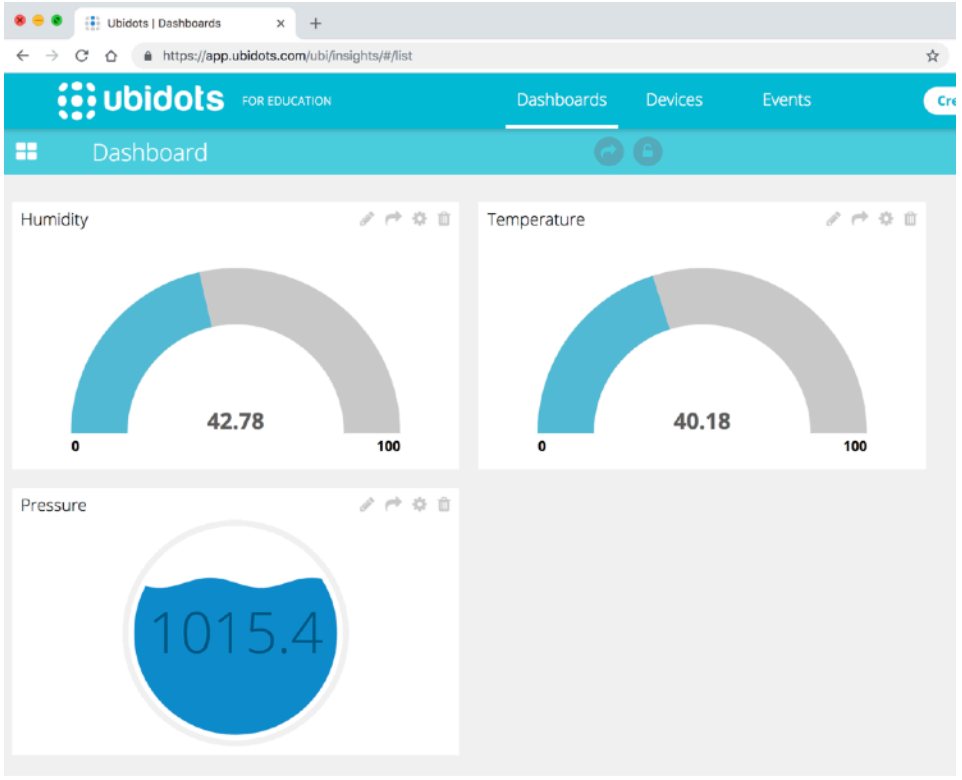
Creating an app 2/2



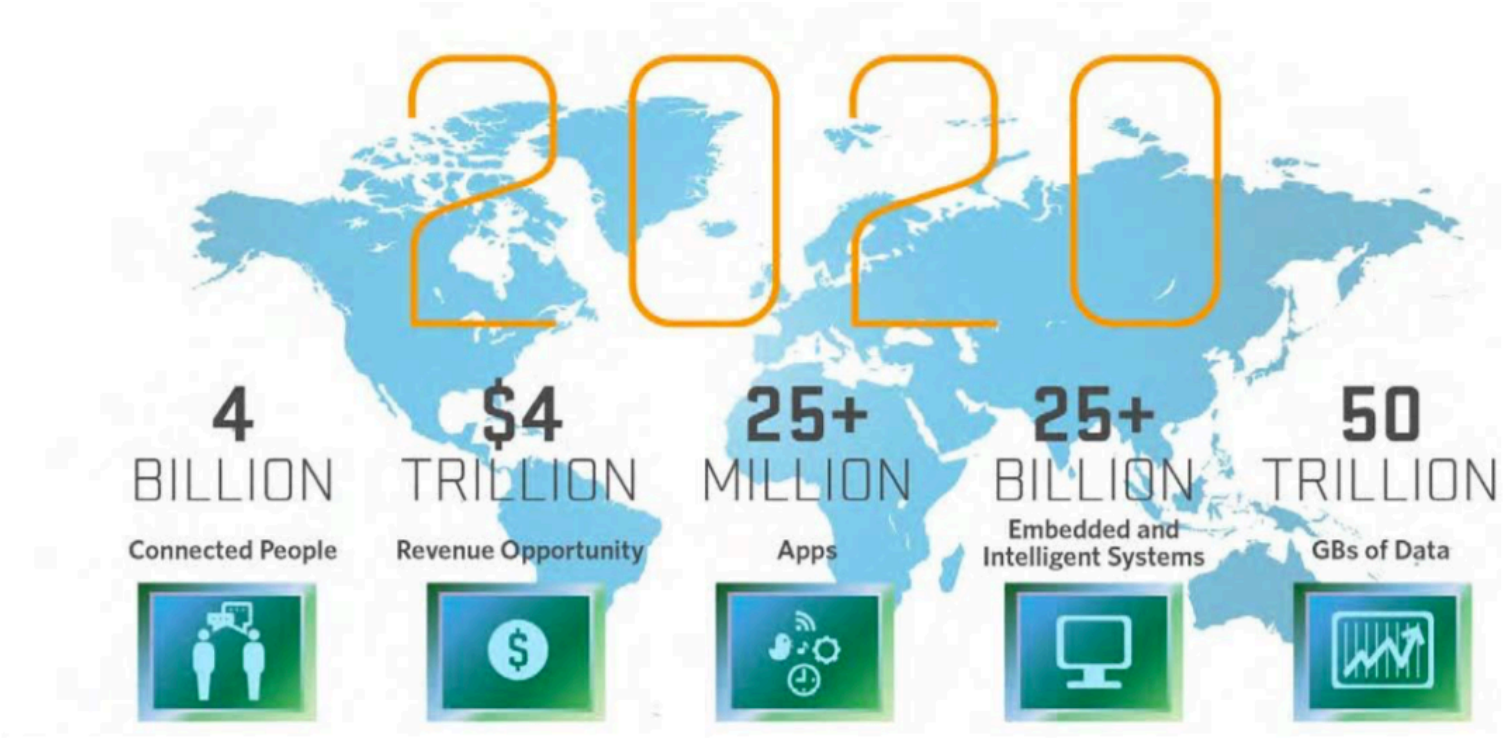
Security, Oauth…

Another example: A meteo station

```
mqttc=mqtt.Client()
mqttc.username_pw_set("A1E-2DvBgZ..........aOcG4SRuTkgH", password=None)
mqttc.connect("things.ubidots.com", 1883, 60)


mqttc.loop_start()
mqttc.publish("/v1.6/devices/meteopi/humidity", humi)
mqttc.publish("/v1.6/devices/meteopi/pressure", pres)
mqttc.publish("/v1.6/devices/meteopi/temperature", temp)
mqttc.loop_stop()
```



IoT market estimation worldwide



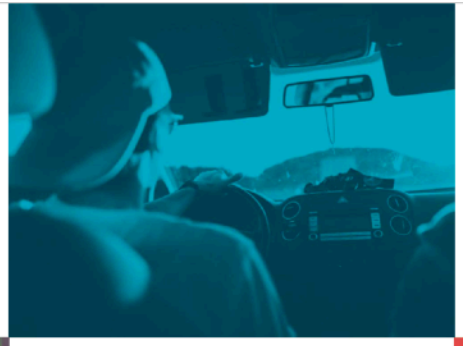
Julio 2017, IDC www.idcspain.com

All big companies are active in this area

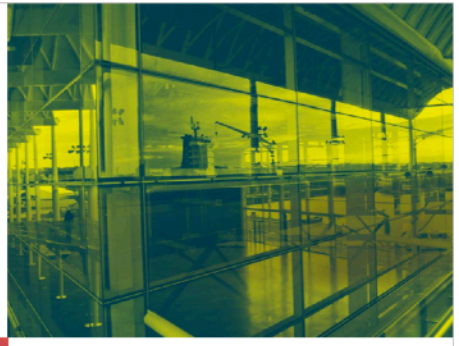
- Telefonica: https://iot.telefonica.com/thinking-things
- CISCO: https://www.cisco.com/c/en/us/solutions/internet-of-things/overview.html
- Google: https://cloud.google.com/solutions/iot/
- HP: https://www.hpe.com/us/en/solutions/internet-of-things.html
- IBM: https://www.ibm.com/internet-of-things/
- Microsoft: http://www.lab-of-things.com/
- Oracle: https://www.oracle.com/solutions/internet-of-things/
- Samsung: http://www.samsung.com/global/business/networks/solutions/solutions/iot-solution
- Apple: https://www.apple.com/shop/accessories/all-accessories/homekit
- …

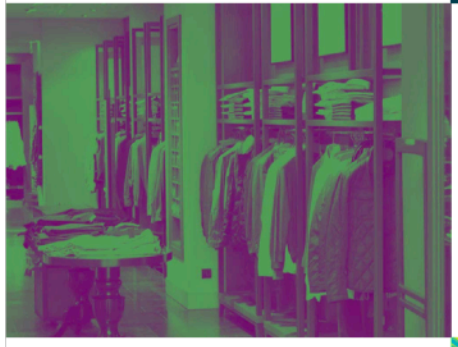
Entornos principales

EL VEHÍCULO CONECTADO
coche conectado, alquiler de coches por hora, leasing avanzado, localización del vehículo, actualización remota de software, etc.



LA INDUSTRIA CONECTADA
gestión de flotas, eficiencia energética, *tracking* de posición y estado de mercancías, gestión de inventario, etc.



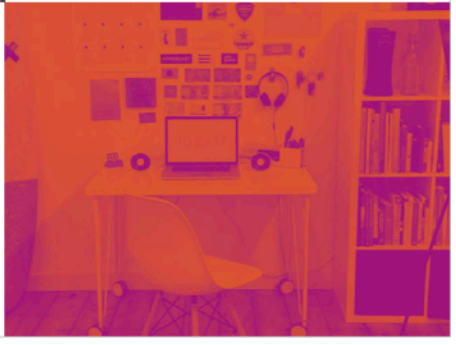


LA TIENDA CONECTADA
localización en espacios cerrados y grandes superficies, ofertas personalizadas en tienda, probadores inteligentes, etc.

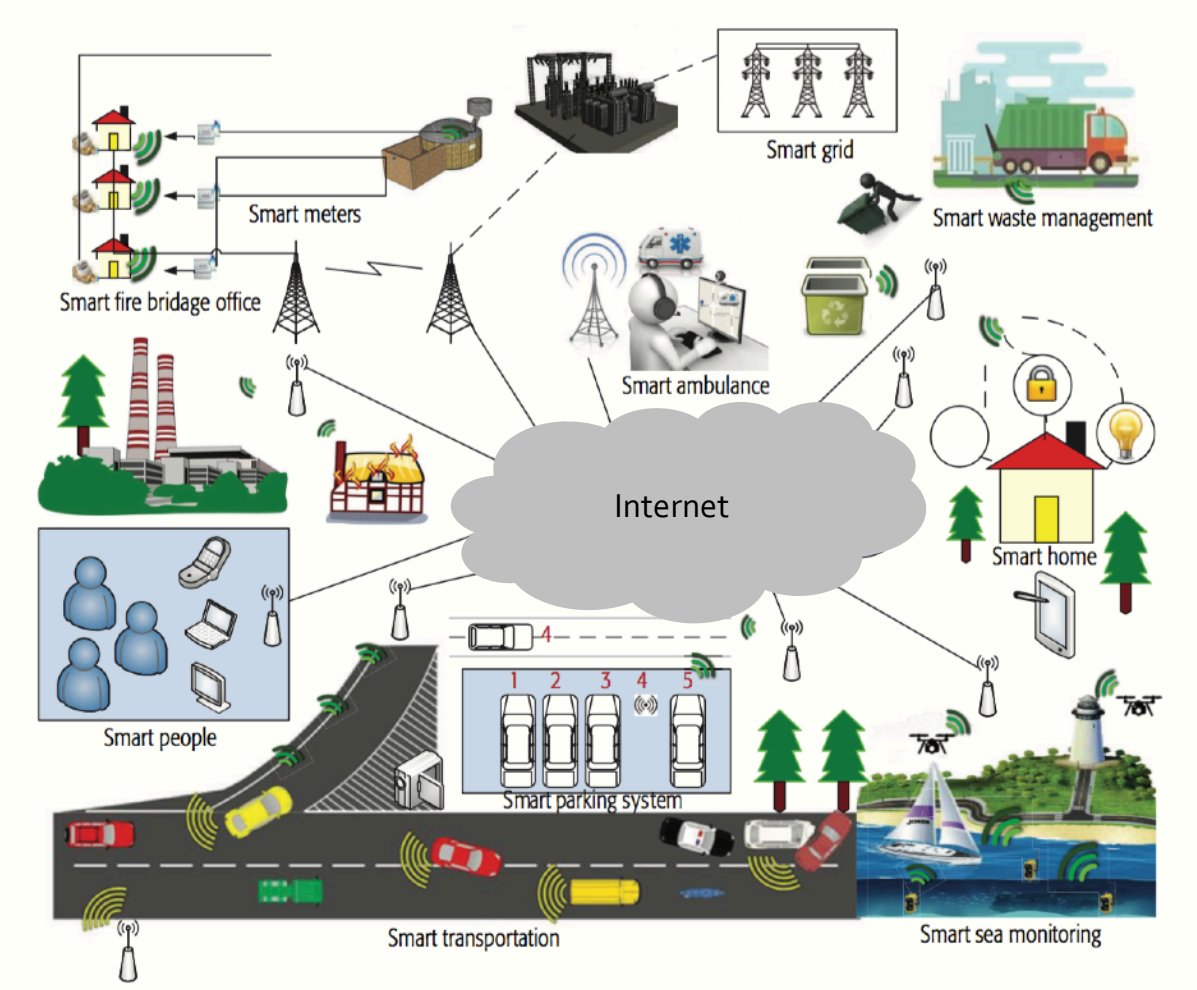
LA CIUDAD CONECTADA
gestión de residuos, *smart parking*, semáforos inteligentes, alumbrado eficiente, etc.

LA PERSONA CONECTADA
wearables, *trackers* de objetos y mascotas, calzado inteligente, ropa inteligente, etc.

EL HOGAR CONECTADO
electrodomésticos conectados, contadores inteligentes, sistema remoto de control de luces, etc.

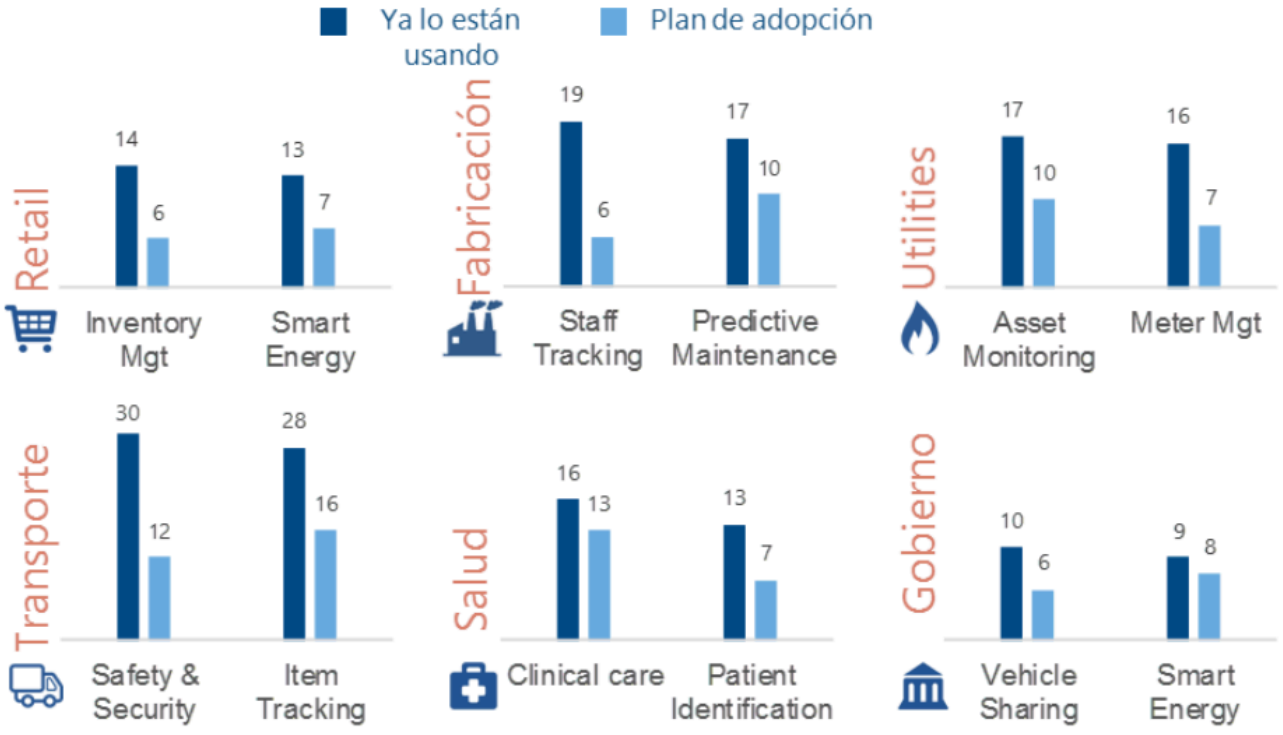


Smarter everything



Enabling Communication Technologies for Smart Cities, IEEE Communications Magazine - January 2017

Casos de Uso de IoT en España

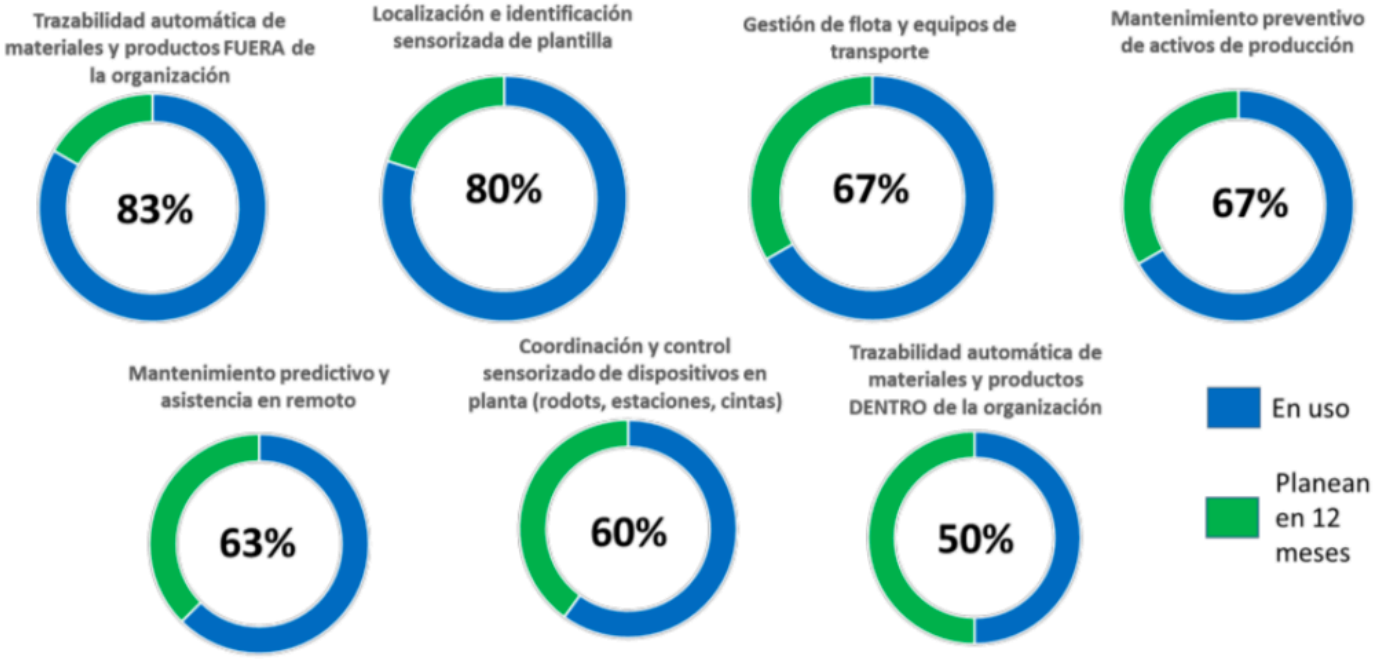


Fuente: IDC - Encuesta IoT en España, 2016

https://www.vodafoneempresa.es/corporate/vbec/files/Informe_IoT_IDC_Vodafone_Business_University.pdf

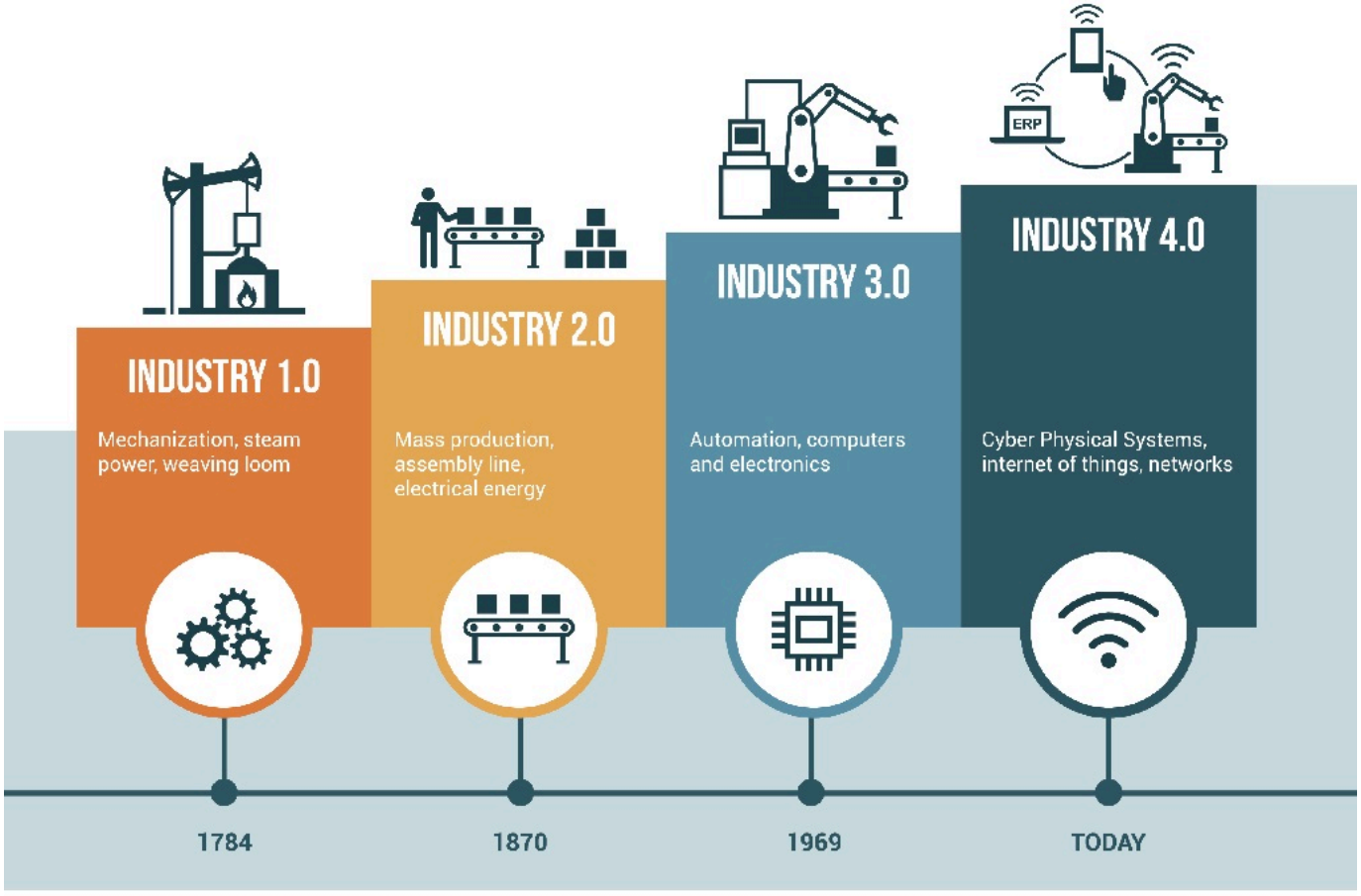
Casos de Uso de IoT en España

Sector Industria



Fuente: IDC - Encuesta IoT en España, 2016

Industry 4.0 and Industrial IoT



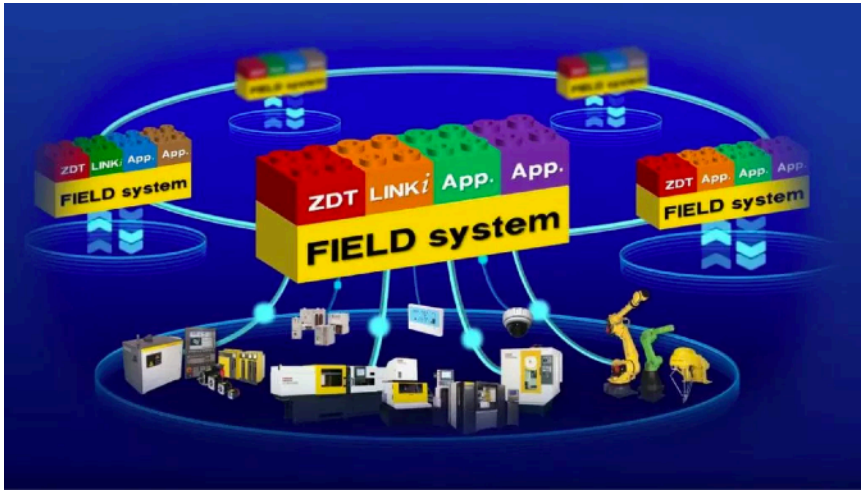
Internet of Robotic Things

- The main difference with the Internet of Things as we know it, is that the devices, the robots, take real action (and are) in the physical world.
- In 2014 ABI Research defined the Internet of Robotic Things or IoRT as:

  "….the concept of the Internet of Robotic Things (IoRT), where intelligent devices can monitor events, fuse sensor data from a variety of sources, use local and distributed intelligence to determine a best course of action, and then act to control or manipulate objects the physical world, and in some cases while physically moving through that world".

Collaborative robots and IoRT

- Example: "FIELD" (FANUC Intelligent Edge Link and Drive)
- Joined effort among **FANUC**, a Japanese and globally active manufacturer of industrial and intelligent robots ad expert in factory automation, **Rockwell Automation**, **Preferred Networks** and **Cisco**
- It uses sensors, middleware, deep learning, edge computing and more to enable industrial robotics devices that coordinate and collaborate.



Internet of things communications models

- In March 2015, the Internet Architecture Board (IAB) released a guiding architectural document for networking of smart objects (RFC 7452) which outlines a framework of four common communication models used by IoT devices.

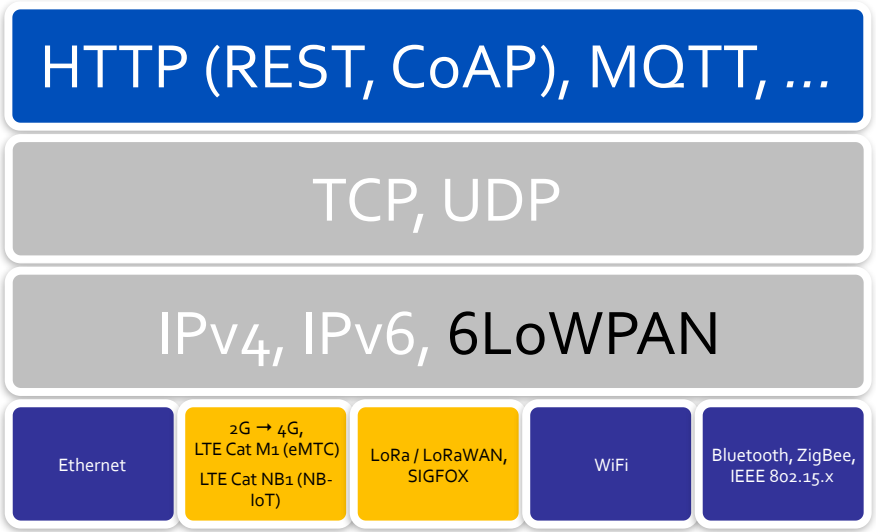
General guidelines

- While developing embedded systems is itself a complex task, designing Internet-connected smart objects is even harder since it requires expertise with Internet protocols in addition to software programming and hardware skills.
- To simplify the development task, and thereby to lower the cost of developing new products and prototypes, **the reuse of prior work is essential.**
- RFC 7452 provides high-level guidance on the use of Internet technology for the development of smart objects, and connected systems in general.

https://tools.ietf.org/html/rfc7452

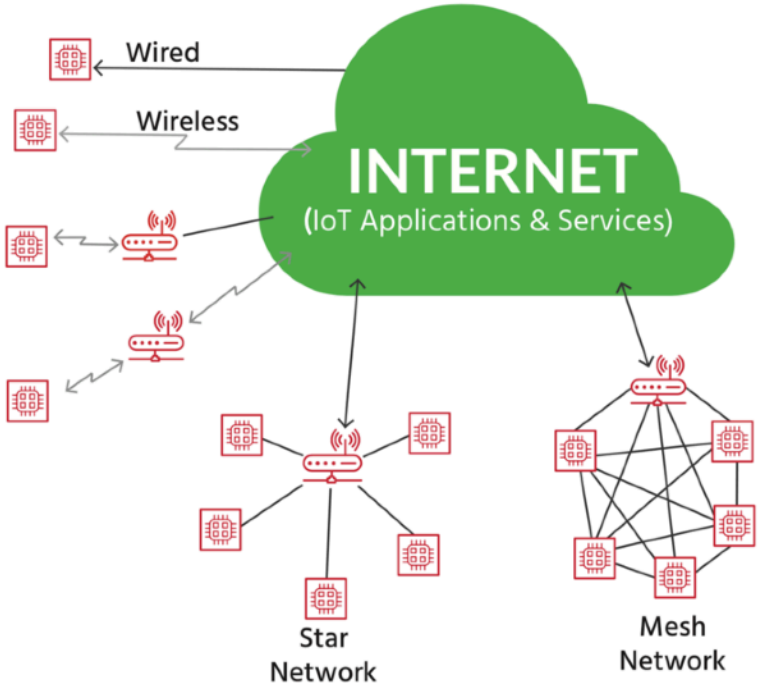
Main guidelines

- Reuse Internet Protocols:
  - Most smart object deployments can make use of the already- standardized Internet Protocol Suite.
  - Internet protocols can be applied to almost any environment due to their generic design and typically offer plenty of potential for reconfiguration, which allows them to be tailored for the specific needs.



Communication patterns

- In the following we will review the communication patterns used in the smart object environment. Namely:
  - Device-to-Device
  - Device-to-Cloud
  - Device-to-Gateway
  - Back-End Data Sharing
- It is possible that more than one pattern can be applied at the same time in a product.



Device-to-Device Communication Pattern

- Figure below illustrates a communication pattern where two devices developed by different manufacturers are desired to interoperate and communicate directly.

- The example above considers a light switch that talks to a light bulb with the requirement that each may be manufactured by a different company (Manufacturer A and B.)
- Other cases can be found with **fitness equipment**, such as heart rate monitors and cadence sensors.

Example

Ellipse

See all reviews

A smart bike lock that connects to your phone to provide keyless entry, theft detection, bike sharing, crash alerts and more.

Quantity

- 1 +

$199

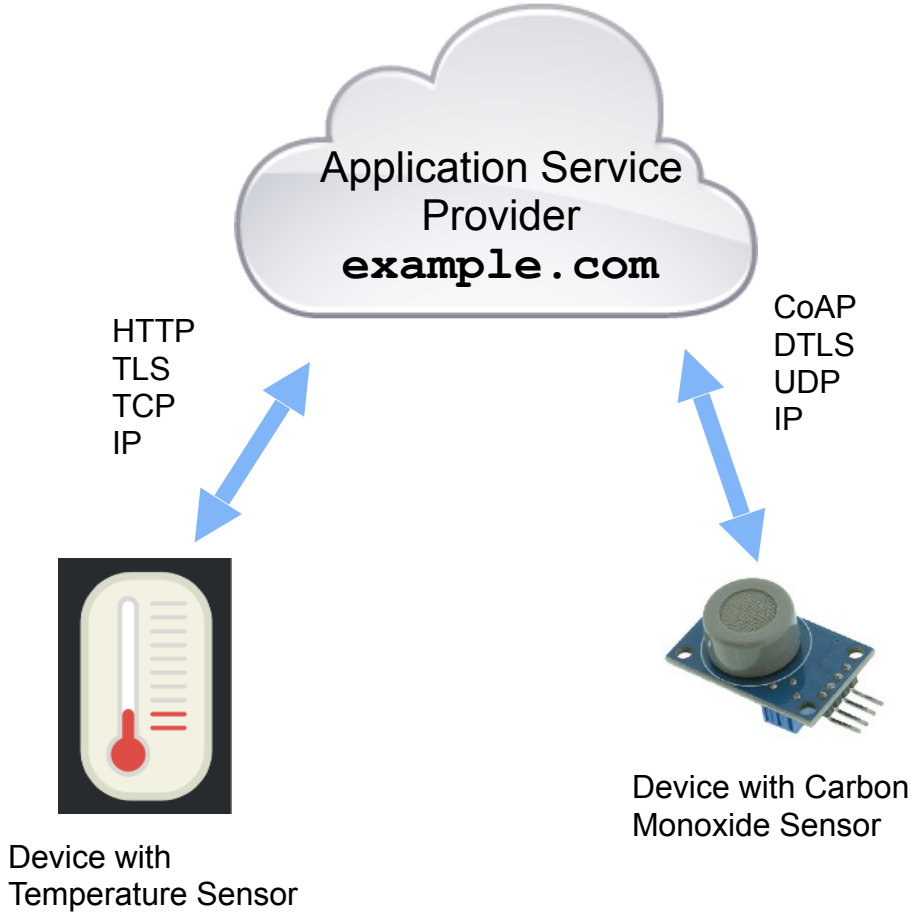
BUY NOW

Free shipping to the U.S.

International shipping starts at $20.

Device-to-Cloud Communication Pattern

- The Figure shows a communication pattern for uploading sensor data to an application service provider.
- Often the application service provider (`example.com` in our illustration) also sells smart objects.
- In that case, the entire communication happens internal to the provider and no need for interoperability arises.



Example....

BLOOM Home Control insecticida volador eléctrico líquido controlable desde el móvil aparato + 1 recambio

35,69 € (35,69 € / Unidad)

UNIDADES

\- 1 +

AÑADIR AL CARRO

Información general

Bloom Home Control es un insecticida eléctrico líquido con la fórmula más avanzada para protegeros a ti y a tu familia de los mosquitos común y tigre.

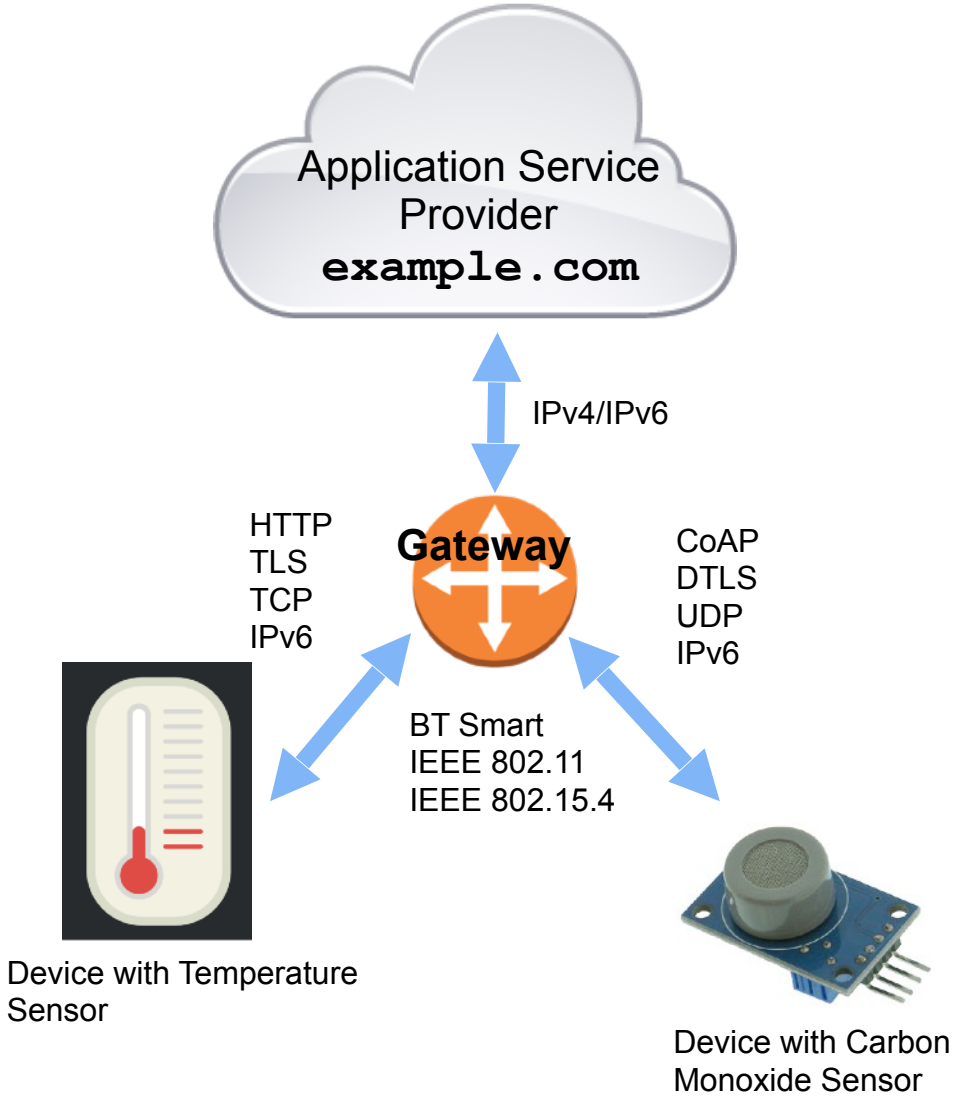
- Programable según tus necesidades.
- Se conecta y controla a distancia a través de tu Smartphone (aplicación disponible para Android & iOS). Cuando lo necesites y desde donde quieras
- Control de la intensidad: Normal y Max.
- Modo Inteligente: este modo ajustará la intensidad y duración de forma automática para ti dependiendo del tamaño de la habitación, hábitos de uso y la intensidad de mosquitos según AccuWeather.
- Recordatorio para comprar un recambio. Recibe una notificación automática cuando el contenido líquido esté por debajo del 20%.

Device-to-Cloud Communication Pattern

- While this pattern allows using IP-based communication end to end, it may still lead to silos.
  - A "silo" describes any system that is unable to operate with any other system, meaning it's closed off from other systems
  - **A frequent concern from end users is that a change in the business model (or bankruptcy) of the IoT device/service provide might make the hardware become unusable.**
- To prevent silos, service providers may allow third-party device vendors to connect to their server infrastructure as well.
  - For those cases, the protocol interface used to communicate with the server infrastructure needs to be made available, and various standards are available, such as CoAP, HTTP, MQTT, UDP, IP, etc.,

Device-to-Gateway Communication Pattern

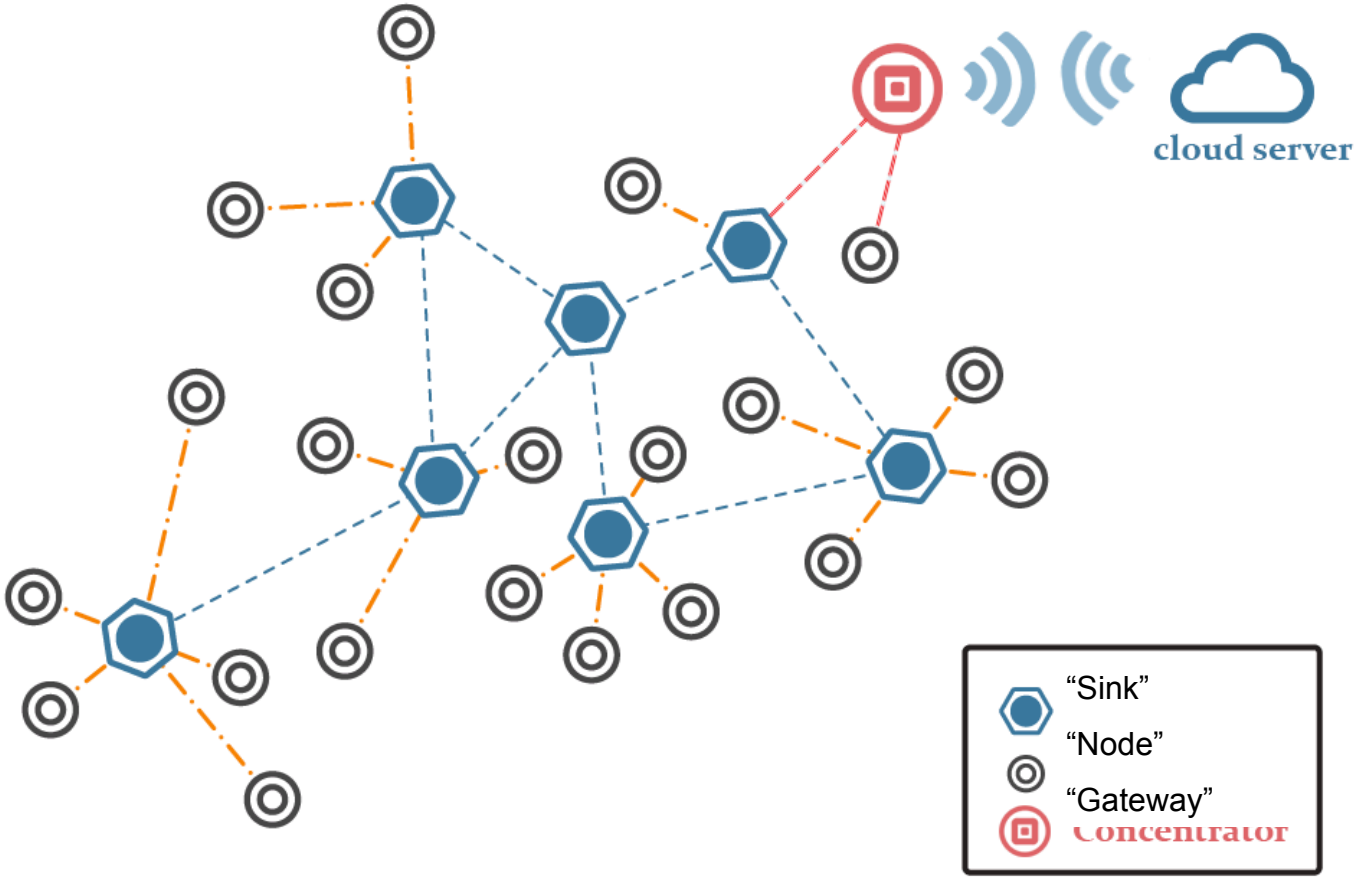
- The device-to-cloud communication pattern, is convenient for vendors of smart objects and works well if they choose a radio technology that is widely deployed in the targeted market, such as Wi-Fi.
- Sometimes, less-widely-available radio technologies are needed (such as IEEE 802.15.4) or special application-layer functionality (e.g., local authentication and authorization) has to be provided or interoperability is needed with legacy, non-IP-based devices.
- **In those cases, some form of gateway has to be introduced into the communication architecture that bridges between the different technologies** and performs other networking and security functionality.



Example...

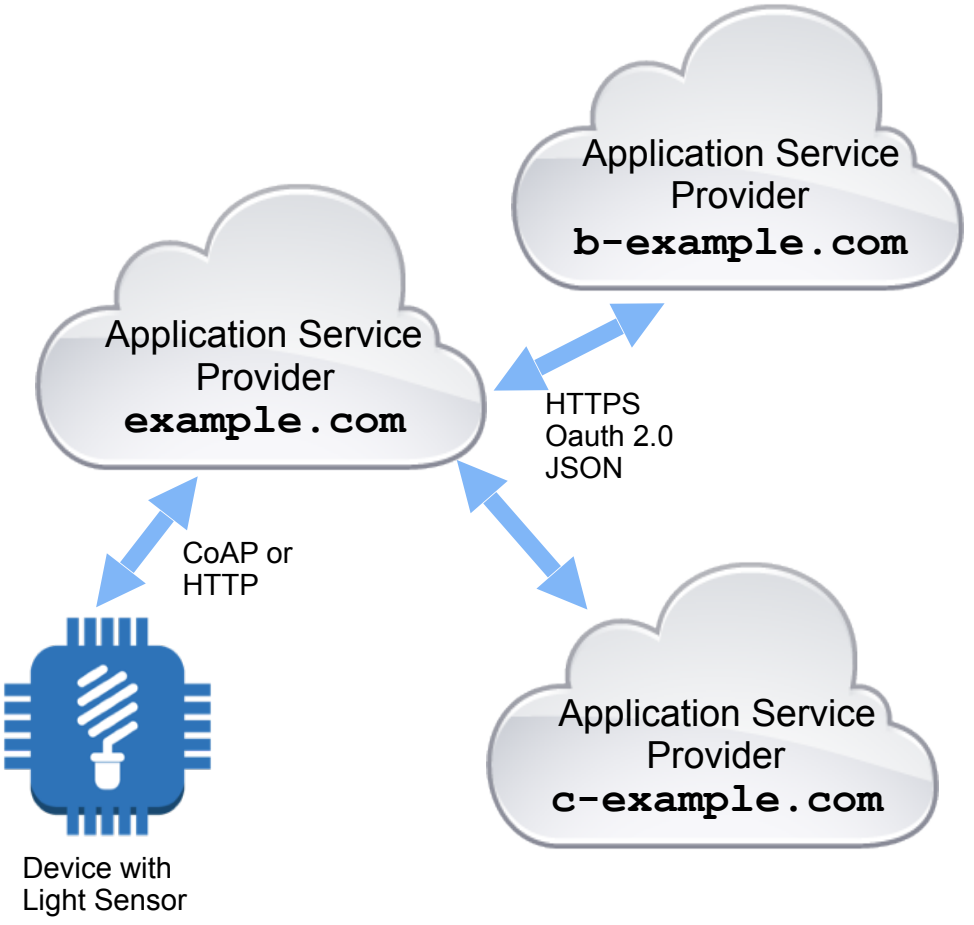

xignal
xignal
xignal Mousetrap
detect → send → report
xignal
xignal

Device-to-Gateway: More complex configurations



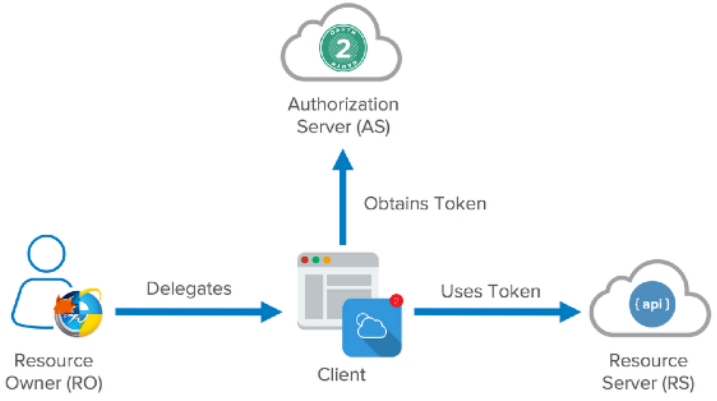
Back-End Data Sharing Pattern

- The device-to-cloud pattern often leads to silos; IoT devices upload data only to a single application service provider.
- However, users often demand the ability to export and to analyse data in combination with data from other sources.
- Hence, the desire for granting access to the uploaded sensor data to third parties arises.



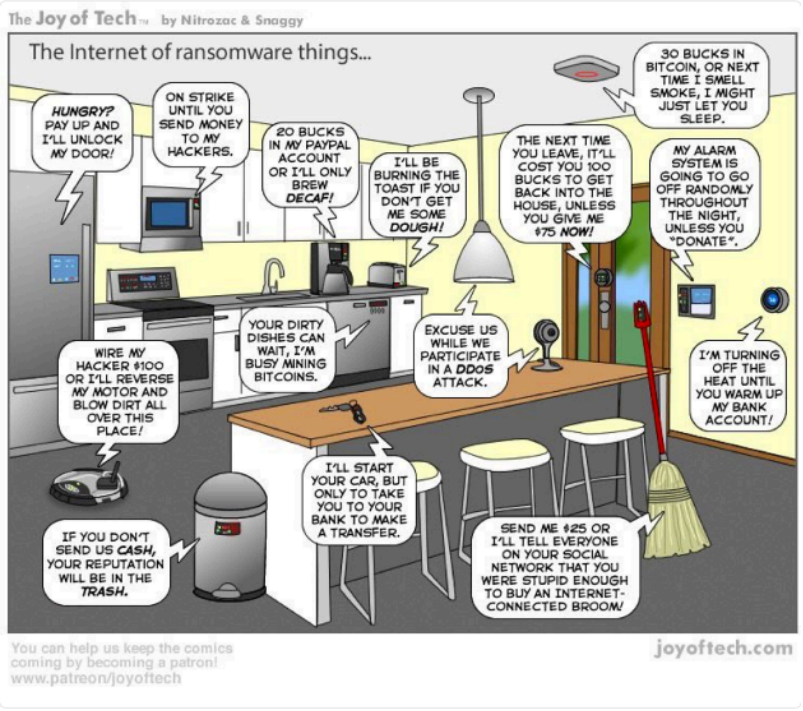
Back-End Data Sharing Pattern

- This pattern is known from the Web in case of mashups and is, therefore, reapplied to the smart object context.
- To offer familiarity for developers, typically a RESTful API design in combination with a federated authentication and authorization technology (**like OAuth 2.0**) is reused.
- While this offers reuse at the level of building blocks, the entire protocol stack (including the information/data model and RESTful Web APIs) is often not standardized.



Security considerations

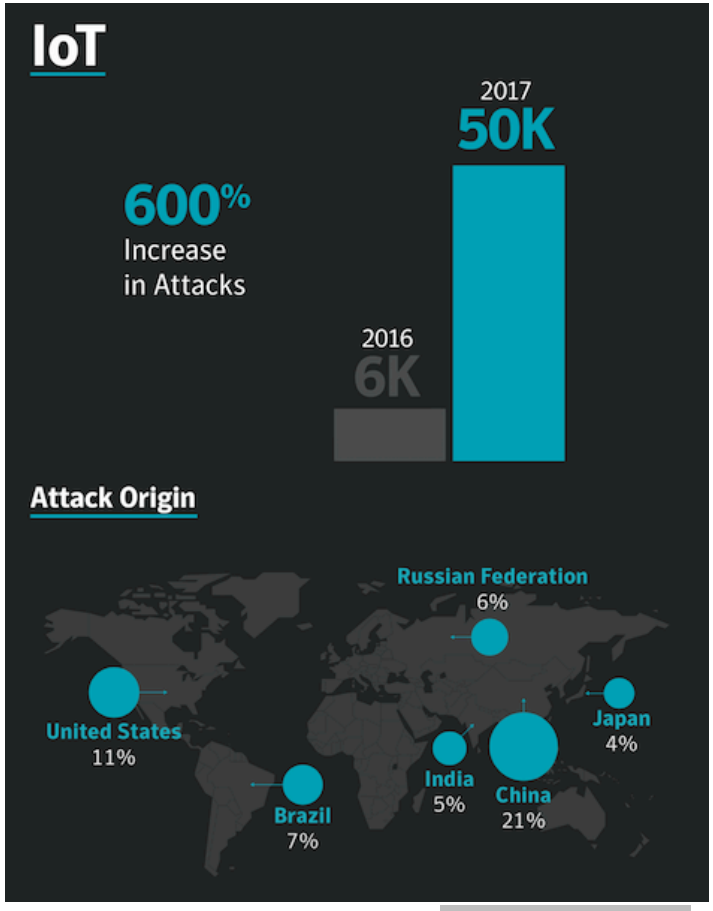
- There is no longer a solid perimeter
  - Wireless, mobile, web sites, computing everywhere
- Multiple vendors providing solutions
- Security is not a selling point - First to market
- Outsourcing
- ~~Example: Dyn Attack~~
  - October 21, 2016
  - https://dyn.com/blog/dyn-analysis-summary-of-friday-october-21-attack/



https://github.com/jgamblin/Mirai-Source-Code

Security considerations

- Security is often even more important for smart objects than for more traditional computing systems, since **smart objects often operate autonomously without any human interaction for a long time period**.
- The problem is compounded by the fact that there are often **fewer resources available** in constrained devices to actually implement security
- A key part of any smart object design is the problem of how to establish trust for a smart object.
  - Trust models should distinguish between devices susceptible to physical compromise and devices with some level of physical security.



Symatec ISTR 2018

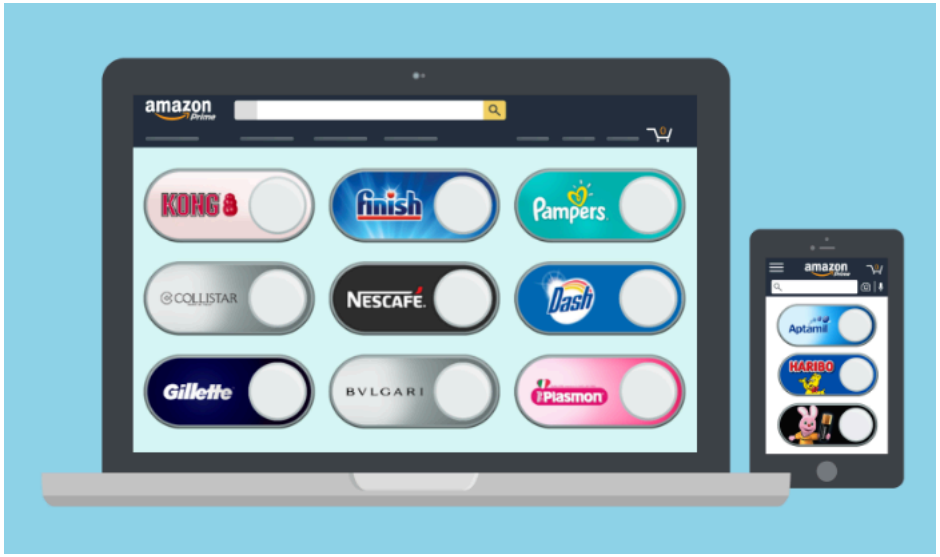
Things

- ✓ Physical things exist in the physical world and are capable of being **sensed, actuated and connected**.
  - Examples of physical things include the surrounding environment, industrial robots, goods and electrical equipment.
- ✓ Virtual things exist in the information world and are capable of being **stored, processed and accessed**.
  - Examples of virtual things include multimedia content and application software.

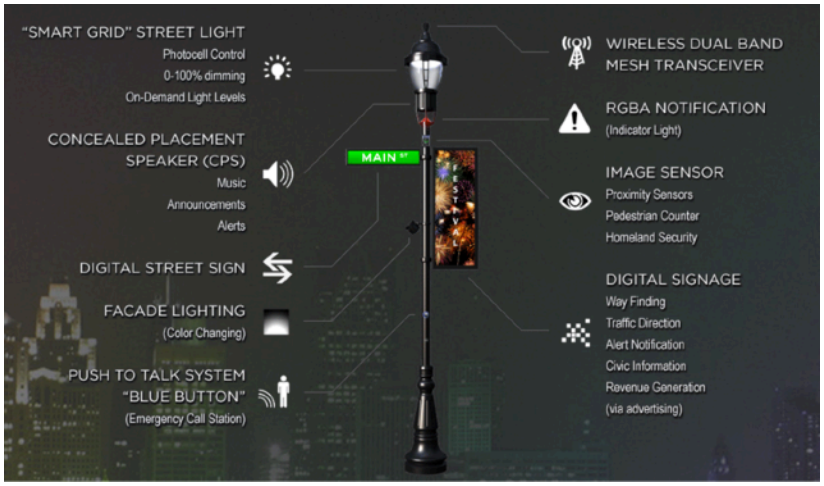
Source: Recommendation ITU-T Y.2060

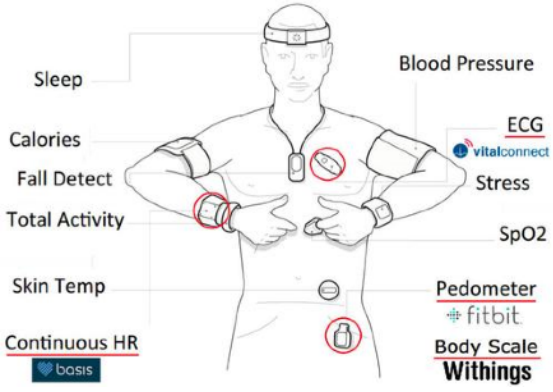
Virtual and Physical example

- Amazon Dash Button



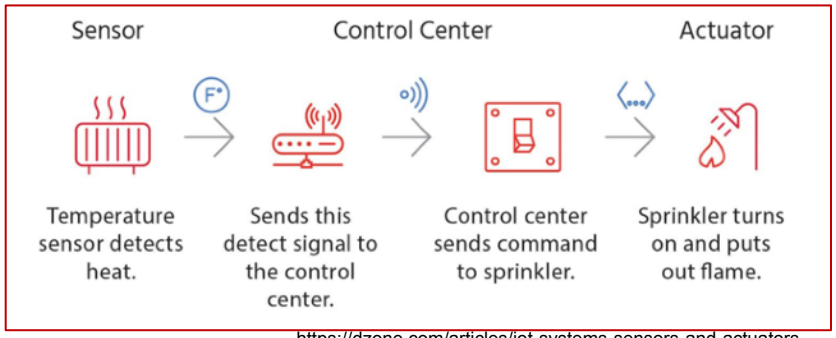
More examples of Things



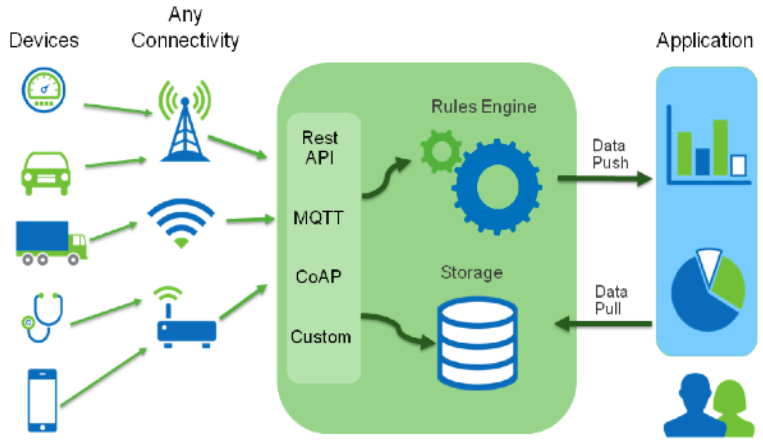


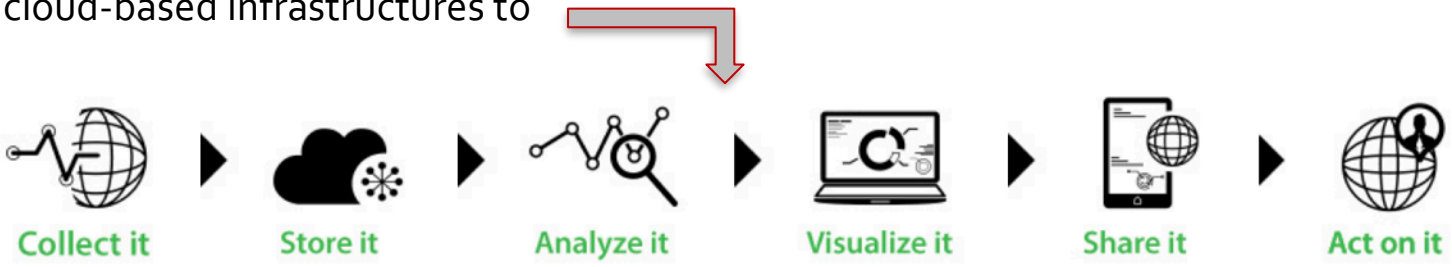
Things… and the rest

- **Devices ("things")**
  - The objects we want to connect. These could be sensors, actuators, robots, cars, whatever can be connected.
  - A lot of inheritance from the world of "**sensors networks**"
- Connectivity
  - You will need to connect the things reliably to the Internet.
  - Wireless connectivity is central to this task
- Platform
  - the collected data needs to be stored and processed somewhere. Typically cloud-based infrastructures to



https://dzone.com/articles/iot-systems-sensors-and-actuators

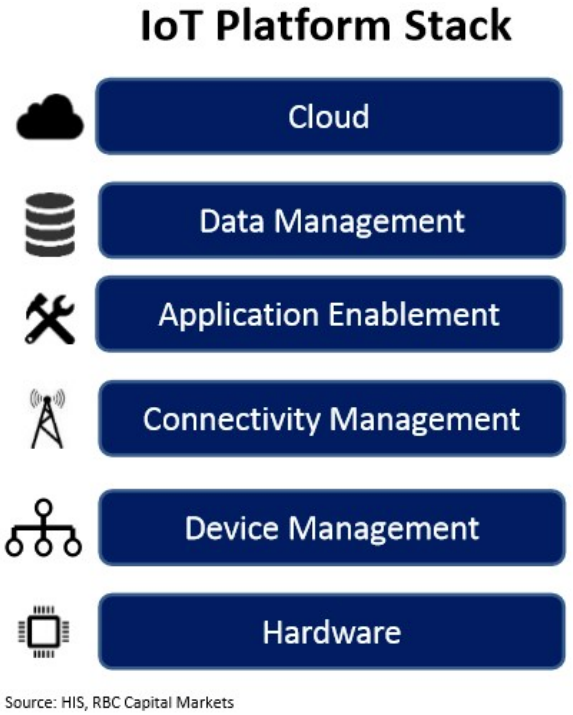




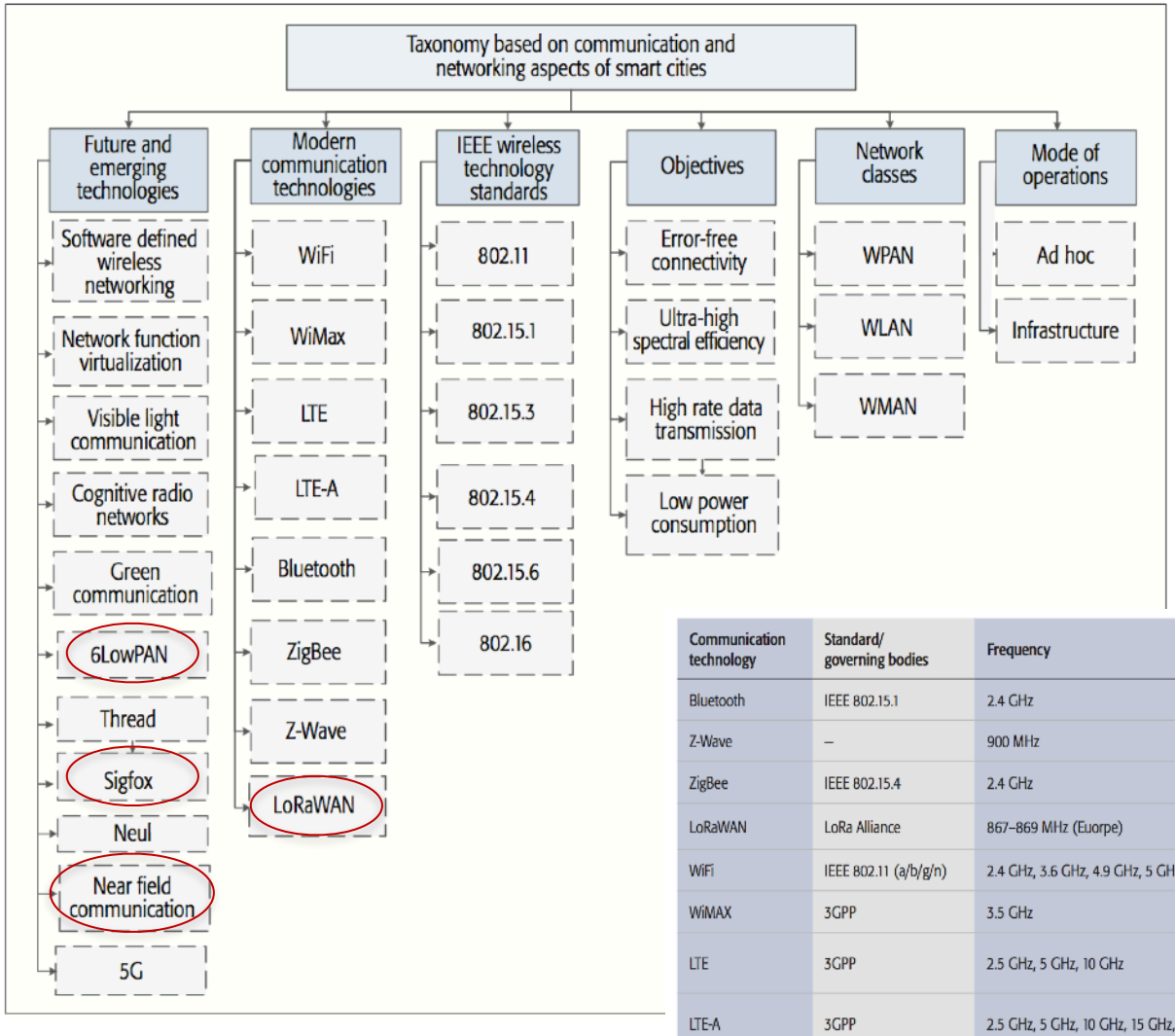
Platforms

- Amazon Web Services
- Google Cloud IoT
- Microsoft Azure IoT Suite
- Salesforce IoT
- Oracle Internet of Things
- Cisco IoT Cloud Connect
- IBM Watson Internet of Things

- FIWARE: https://www.fiware.org/
- ThingSpeak: https://thingspeak.com/
  - Based on MATLAB
- Firebase: https://firebase.google.com/
  - http://ozgur.github.io/python-firebase/
- GroveStreams: https://grovestreams.com/
- SensorCloud: http://www.sensorcloud.com/
- Carriots: https://www.carriots.com/
- Ubidots: https://app.ubidots.com/accounts/signin/
- ...



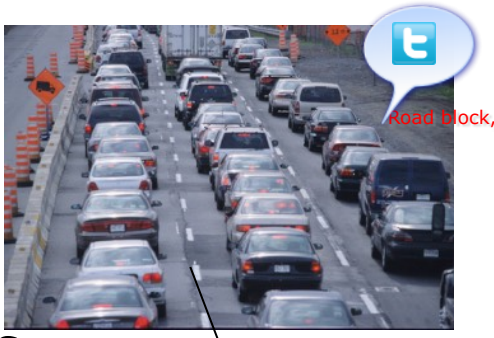
Enabling communication and networking technologies

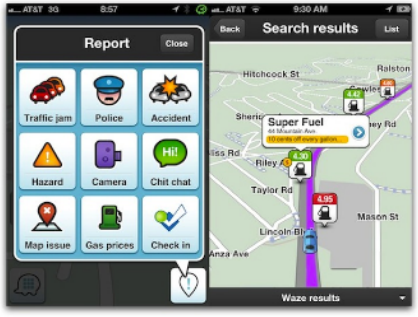


Communication technology	Standard/ governing bodies	Frequency	Range (approximately)	Data rates	Topology
Bluetooth	IEEE 802.15.1	2.4 GHz	1–100 m	1 Mb/s	Point-to-point
Z-Wave	–	900 MHz	100 m	9.6–100 kb/s	Star, cluster, mesh
ZigBee	IEEE 802.15.4	2.4 GHz	10–20 m	25 0kb/s	Mesh
LoRaWAN	LoRa Alliance	867–869 MHz (Euorpe)	2–5 km	290 b/s–50 kb/s	Star
WiFi	IEEE 802.11 (a/b/g/n)	2.4 GHz, 3.6 GHz, 4.9 GHz, 5 GHz, 5.9 GHz	100 m	1–5 4Mb/s	Star
WiMAX	3GPP	3.5 GHz	50 km	75 Mb/s	Point-to-multipoint, mesh
LTE	3GPP	2.5 GHz, 5 GHz, 10 GHz	30 km	30 0Mb/s(DL), 75 Mb/s(UL)	Star
LTE-A	3GPP	2.5 GHz, 5 GHz, 10 GHz, 15 GHz, 20 GHz	30Km	1Gb(DL), 500Mb/s(UL)	Point-to-point

Beyond conventional sensors

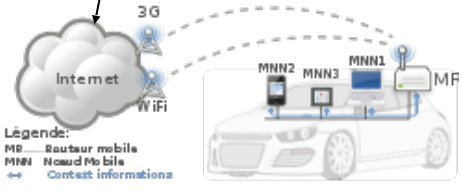
- Human as a sensor (citizen sensors)
  - e.g. tweeting real world data and/or events
- Virtual (software) sensors
  - e.g. Software agents/ services generating/ representing data





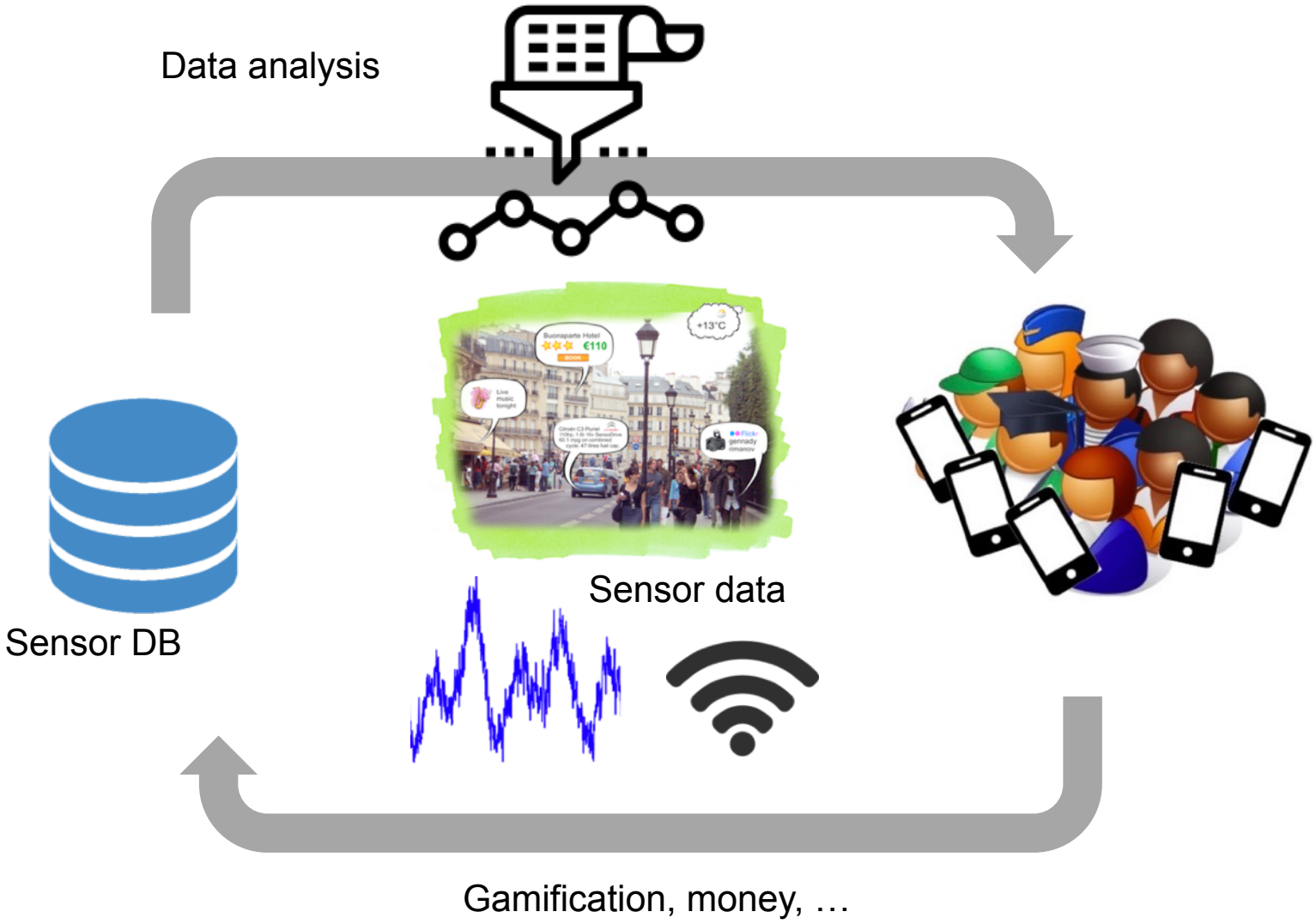
https://www.waze.com

Road block, A3

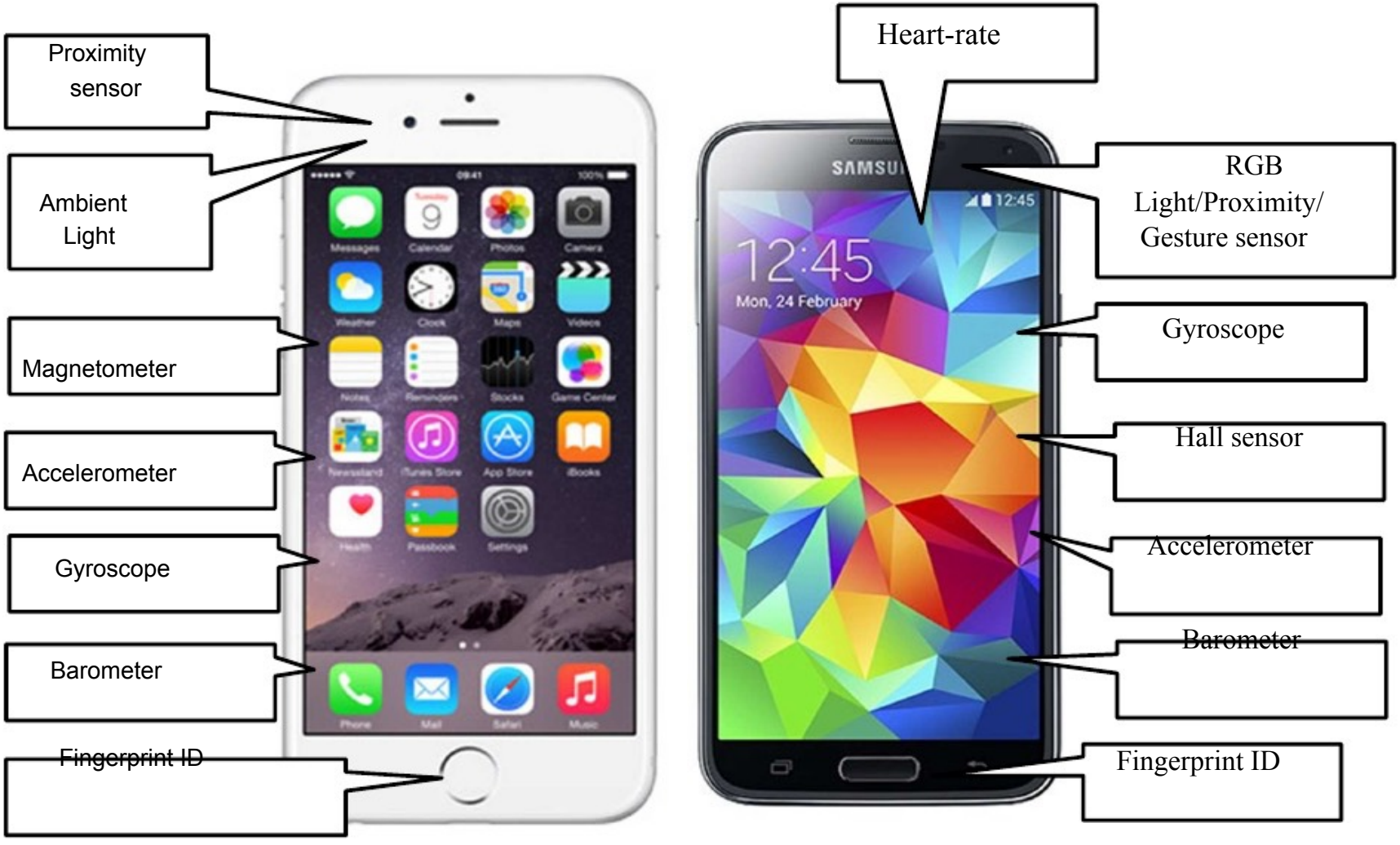


Suggest a different route

Crowdsensing



Sensors in Modern Smart Phones



Voice based interfaces: smart speakers

Amazon Echo, is a “smarthome hub”, essentially a speaker that can listen to users and respond to commands, it can play music, answer questions, read audiobooks, deliver traffic and weather reports, control lights and thermostats, order pizza, order an Uber and much more. The company has also released a similar, lower-priced device called the Tap.

Alexa is Amazon’s cloud-based voice service available on tens of millions of devices from Amazon and third-party device manufacturers. With Alexa, you can build natural voice experiences that offer customers a more intuitive way to interact with the technology they use every day. Our collection of tools, APIs, reference solutions, and documentation make it easy for anyone to build with Alexa.

Google Home speakers enable users to speak voice commands to interact with services through Google's intelligent personal assistant called Google Assistant. A large number of services, both in-house and third-party, are integrated, allowing users to listen to music, control playback of videos or photos, or receive news updates entirely by voice

Open Data

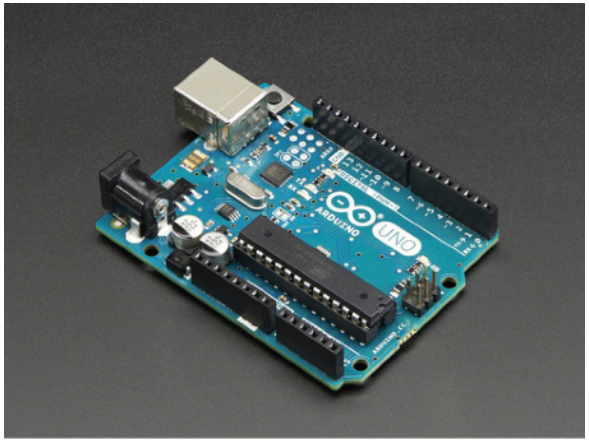
- "Cities today are vast repositories of information, endlessly collecting and archiving data. When semantically organized, the data can be exposed, shared, and interconnected. Giving people the right kind of access to this information can spark new applications and services, new ways of living, creating and being."
- Global movement to open up public data sets to make public data more accessible
  - Sparks innovation: Creation of apps and services
  - Greater transparency in government
- Big Data techniques!

Berners Lee, "The year open data went worldwide", TED talks: http://www.ted.com/talks/tim_berners_lee_the_year_open_data_went_worldwide.html

Some example of open data projects

- http://data.worldbank.org/
- http://opendataday.org/
- http://datos.gob.es/
- https://data.baltimorecity.gov
- http://vancouver.ca/your-government/open-data-catalogue.aspx
- https://data.smartdublin.ie/
- https://data.rennesmetropole.fr/page/home/
- https://data.grandlyon.com/
- http://opendata.paris.fr
- http://opendata.comune.fi.it
- http://dati.comune.roma.it
- http://data.london.gov.uk
- http://data.gov.uk
- ...

Things: HW



https://www.postscapes.com/internet-of-things-hardware/

The 3 stages of an IoT journey

"The minimum viable product (MVP) is that version of a new product which allows a team to collect the maximum amount of validated learning about customers with the least effort." by Eric Ries

Proof of Value
(Engineering Samples)

Build an MVP to measure the ROI and to validate the value proposition interest

Pilot
(Pilot Run)

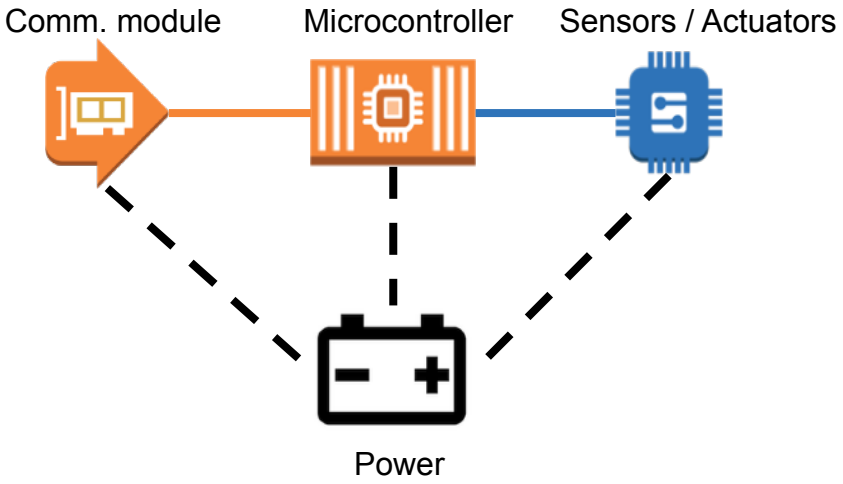
Build a first iteration of the platform to confirm the business case at scale and the functional requirements

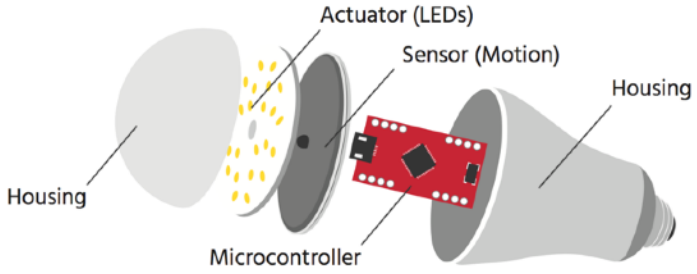
Production
(Mass Production)

Build a scalable and easily manageable platform that is loosely coupled to device manufacturers and protocols

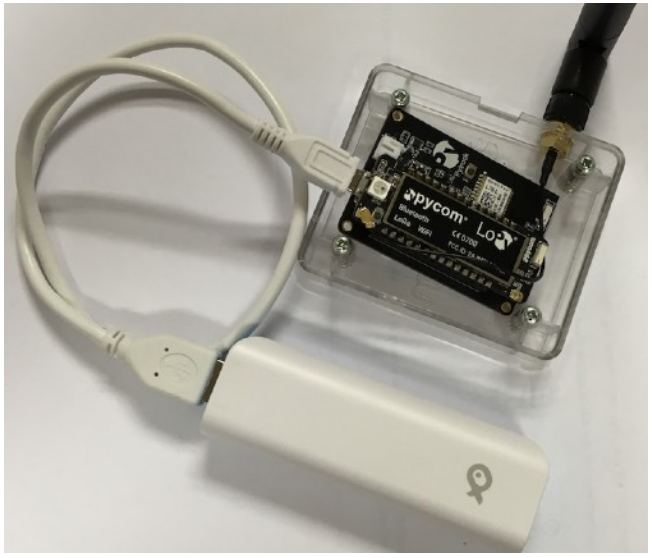
Things

- A "thing" generally consists of **four main parts:**
  - Sensors & actuators
  - Microcontroller
  - Communication unit
  - Power supply
- A "thing" has the **following properties**:
  - It's usually powered by battery. This implies limited source of energy.
  - It's generally small in size and low in cost. This limits their computing capability.



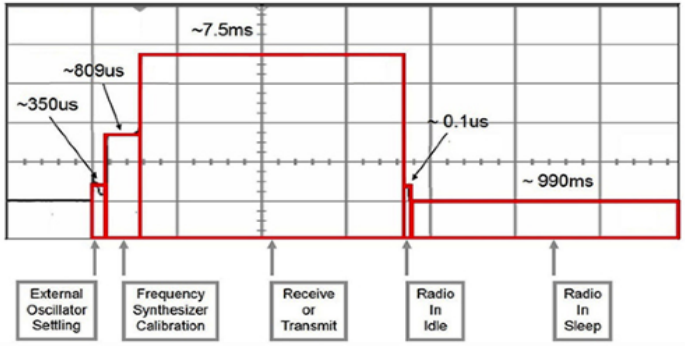


A Reference Guide to the Internet of Things Copyright © 2017 Bridgera LLC, RIoT



Communication Device

- Of the hardware components of a "thing", the radio is usually the most power-consuming component. Compared to the power consumption of the microcontroller or the sensors, the radio transceiver often uses ten times as much power.
- This is due to the processing required for modulating and demodulating the radio signal.
  - For low-power radios, only a small portion of the power consumption is used to send the radio signal into the air.
  - ➔ listening is as power consuming as sending.

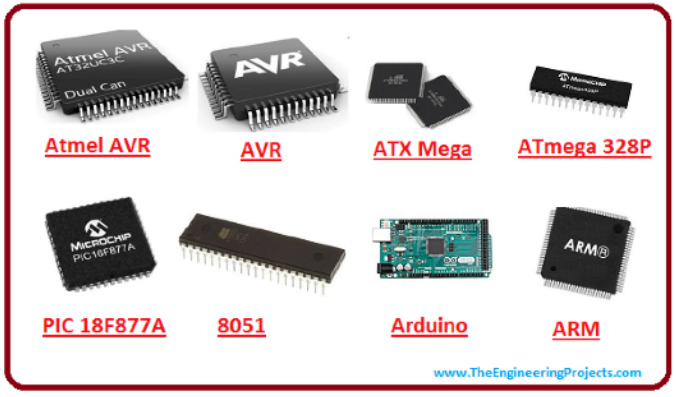


Communication device

- Because the radio is the most power-consuming component, and because idle listening is as expensive as sending data, the radio must be switched off to conserve power.
  - …. But, when the radio is switched off, it is clearly not able to receive any data.
- The radios of the devices in the network must somehow be "synchronized" to find the best compromise between saving energy and sending/receiving

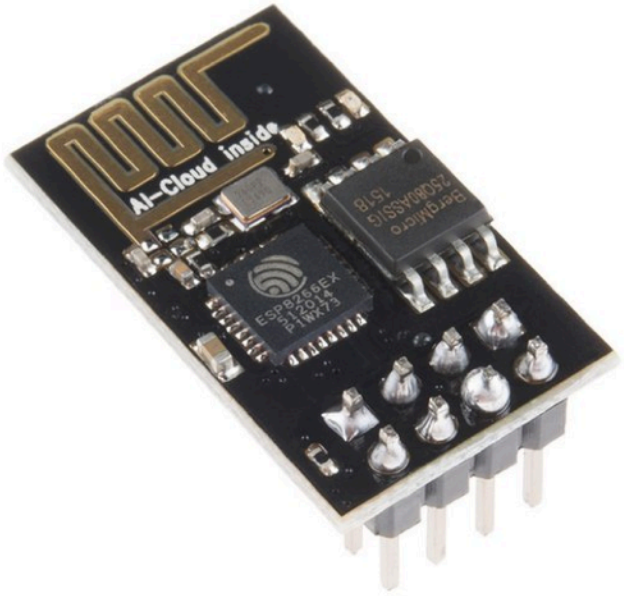
Microcontroller

- The microcontroller gives "things" their intelligence. It runs the software of the devices and is also responsible for connecting the radio with the sensors and actuators.
- A microcontroller is a microprocessor with built-in memory, timers, and hardware for connecting external devices such as sensors, actuators, and radio transceivers.
- Typically, a smart object microcontroller has a few kilobytes of on-chip memory and is run at a clock speed of a few megahertz.



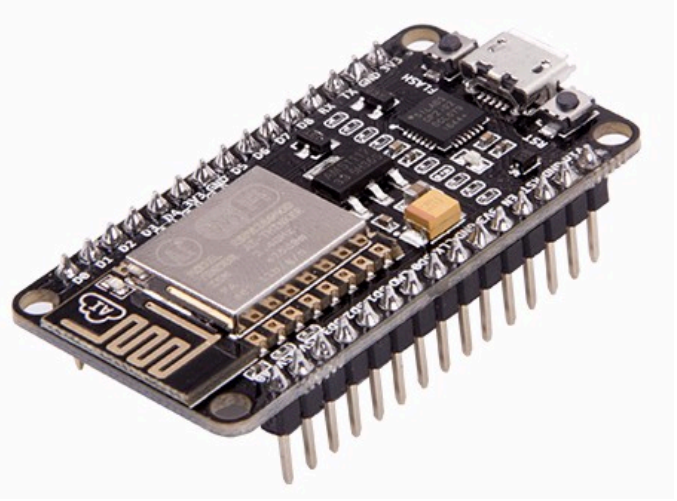
Microcontroller: ESP8266

- The ESP8266 is a low-cost Wi-Fi microchip with full TCP/IP stack and microcontroller capability produced by Shanghai-based Chinese manufacturer, Espressif Systems.
  - http://www.esp8266.com
- Features ESP-01 wireframe.
  - Processor: L106 32-bit RISC microprocessor core based on the Tensilica Xtensa Diamond Standard 106Micro running at 80 MHz
  - 32 KB of instruction RAM, 80 KB of data RAM
  - External QSPI flash: up to 16 MiB is supported (512 KiB to 4 MiB typically included)
  - IEEE 802.11 b/g/n Wi-Fi
    - WEP or WPA/WPA2 authentication, or open networks
  - ...



Microcontroller: ESP8266

- There are many boards based on the ESP8266 chip.
- The figure on the right is of a NodeMCU
- This board uses NodeMCU firmware with Lua as the programming language.
  - http://www.nodemcu.com/

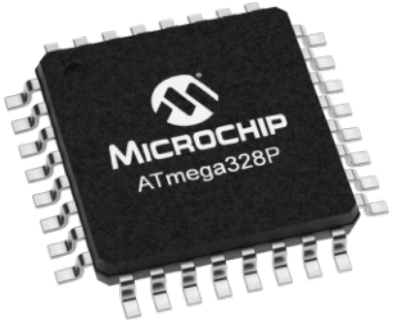


Source: http://www.seeedstudio.com/depot/NodeMCU-v2-Lua-based-ESP8266-development-kit-p-2415.html

Microcontroller: ATmega328P

- The Microchip picoPower 8-bit AVR RISC-based microcontroller combines:
  - 32KB ISP flash memory with read-while-write capabilities,
  - 1024B EEPROM,
  - 2KB SRAM,
  - 23 general purpose I/O lines,
  - 32 general purpose working registers,
  - …

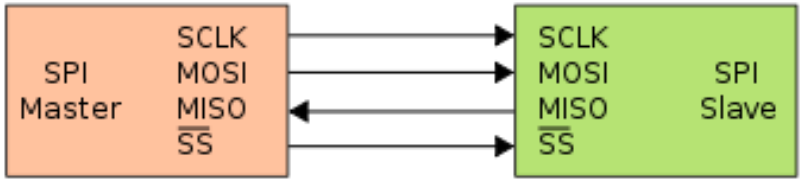
AVR is a family of microcontrollers developed by Atmel beginning in 1996. These are modified Harvard architecture 8-bit RISC single-chip microcontrollers.



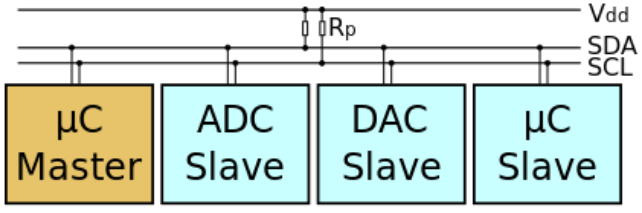
http://www.microchip.com/wwwproducts/en/ATmega328p

Attaching short-distance peripheral

- Serial Peripheral Interface bus (SPI)
  - synchronous serial communication interface
  - SPI devices communicate in full duplex mode using a master-slave architecture with a single master.



- I²C (Inter-Integrated Circuit)
  - multi-master, multi-slave, packet switched, single-ended, serial computer bus



Power sources

- Today, the **most common power source is a battery**, but there are several other possibilities for power, such as **solar cells, piezoelectricity, radio-transmitted energy, and other forms of power scavenging.**
- Rechargeable batteries are not particularly well-suited to smart objects.
  - Instead of using rechargeable batteries, battery-equipped smart objects are typically designed so a single battery should last the entire lifetime of the smart object.

LiPo

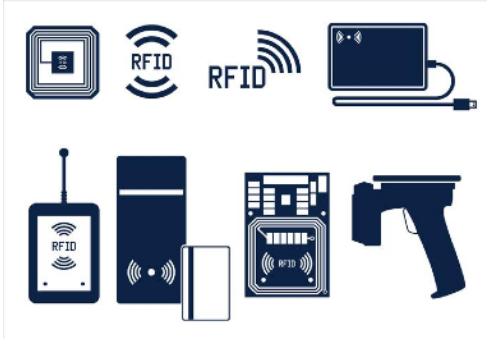
LiPo

Li-ion

Lithium Ion (Li-ion)
Lithium-Ion Polymer (Li-Po)

Power sources

- Power scavenging is a technique that harvests power from the physical environment.
  - **Solar cells** represent the most common form of power scavenging..
  - **Piezoelectricity** is another source for power scavenging. For this source, physical movement is converted into energy used to power the smart object.
    - For example, EnOcean's smart light switches are completely driven by the energy harvested from the act of pressing the light switch.
- **The energy in radio waves can also be used as a power source.**
  - A well-known example of this are Radio Frequency Identification (RFID) tags that use radio energy to power a radio transceiver for a short while.



Operating Systems

- Operating systems for smart objects are tailored to the specific requirements of smart objects and to the specific constraints imposed by the hardware.
  - E.g., Smart object operating systems do not have a user interface; no user directly interacts with the smart object operating system.
- The memory constraints make the programming model different from general purpose operating systems.
- The three most widely used three examples of operating systems are: **Contiki, TinyOS, and FreeRTOS.**

FreeRTOS

- The FreeRTOS kernel is a real time operating system (or RTOS), and an extended solution for microcontrollers and small microprocessors.
- It has a minimal ROM, RAM and processing overhead. Typically an RTOS kernel binary image will be in the region of 6K to 12K bytes.
- Is very simple - the core of the RTOS kernel is contained in only 3 C files. The majority of the many files included in the .zip file download relate only to the numerous demonstration applications.
- https://www.freertos.org/
- Recently bought by Amazon: https://aws.amazon.com/freertos/

TinyOS - programming

- TinyOS is an open source, BSD-licensed operating system designed for low-power wireless devices, such as those used in sensor networks .
- TinyOS applications are developed using nesC, a dialect of the C language that is optimized for the memory limits of sensor networks.
- TinyOS is completely non-blocking:
  - All I/O operations that last longer than a few hundred microseconds are asynchronous and have a callback.

- TinyOS home page: https://github.com/tinyos
- TinyOS tutorial:

  http://tinyos.stanford.edu/tinyos-wiki/index.php/TinyOS_Tutorials

Contiki

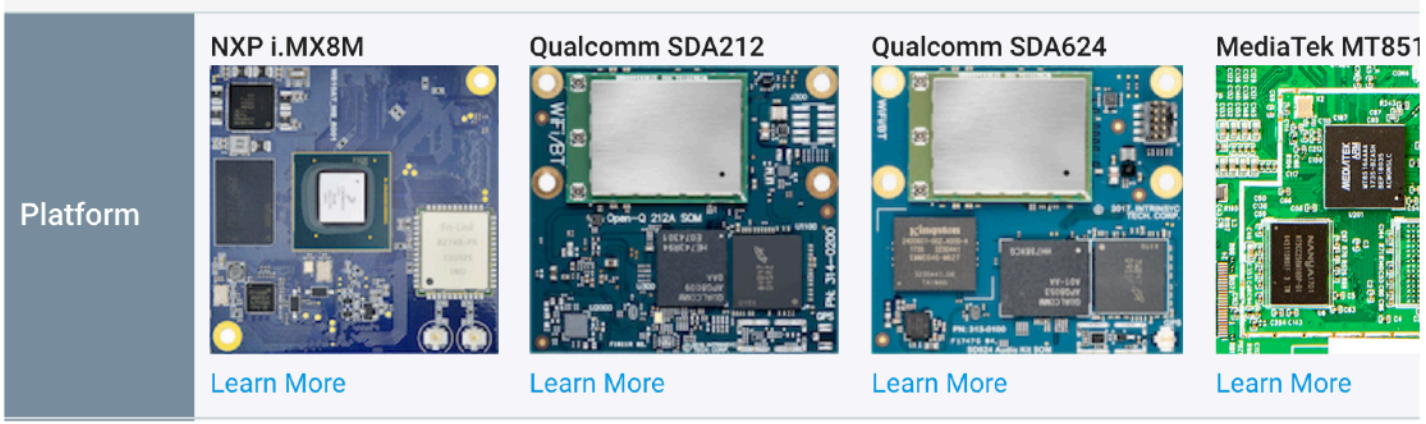
- Contiki is a open source operating system for the Internet of Things.
  - http://www.contiki-os.org/ => http://contiki-ng.org/
    - http://www.contiki-os.org/start.html
  - runs on networked embedded systems and wireless sensor networks.
- It is designed for microcontrollers with small amounts of memory.
  - A typical Contiki configuration is 2 kilobytes of RAM and 40 kilobytes of ROM.
- Contiki provides IP communication, both for IPv4 and IPv6.
  - It has an IPv6 stack that, combined with power-efficient radio mechanisms such as ContikiMAC, allow battery-operated devices to participate in IPv6 networking.
- Contiki supports 6lowPAN header compression, IETF RPL IPv6 routing, and the IETF CoAP application layer protocol.

MicroPython

- MicroPython is a lean and efficient implementation of the Python 3 programming language optimised to run on microcontrollers and in constrained environments.
  - It is compact enough to fit and run within just 256k of code space and 16k of RAM.
  - It is packed full of advanced features such as an interactive prompt, arbitrary precision integers, closures, list comprehension, generators, exception handling and more.
  - **In addition to implementing a selection of core Python libraries, MicroPython includes modules such as "machine" for accessing low-level hardware.**
- You get an interactive prompt (the REPL) to execute commands immediately, along with the ability to run and import scripts from the built-in filesystem.
- MicroPython strives to be as compatible as possible with normal Python (known as CPython) so that if you know Python you already know MicroPython.
- https://micropython.org/
- Use MicroPython online: https://micropython.org/unicorn/

Android Things

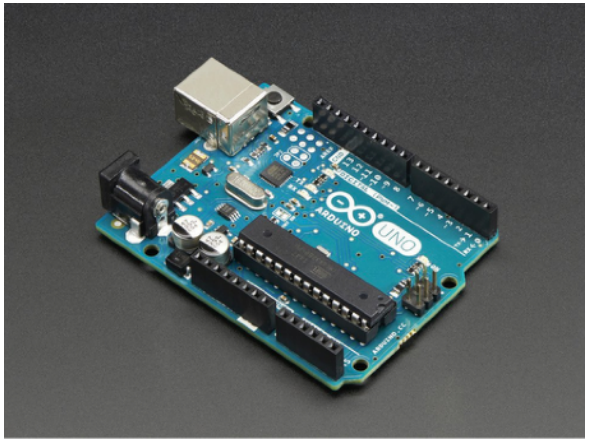
- "Android Things lets you build connected devices for a wide variety of consumer, retail, and industrial applications."
  - Previously known as "Project Brillo"
  - https://developer.android.com/things/



- Weave is a communications platform for Internet of Things (IoT) devices.
  - https://developers.nest.com/weave/
  - Currently used by NEST

nest Developers

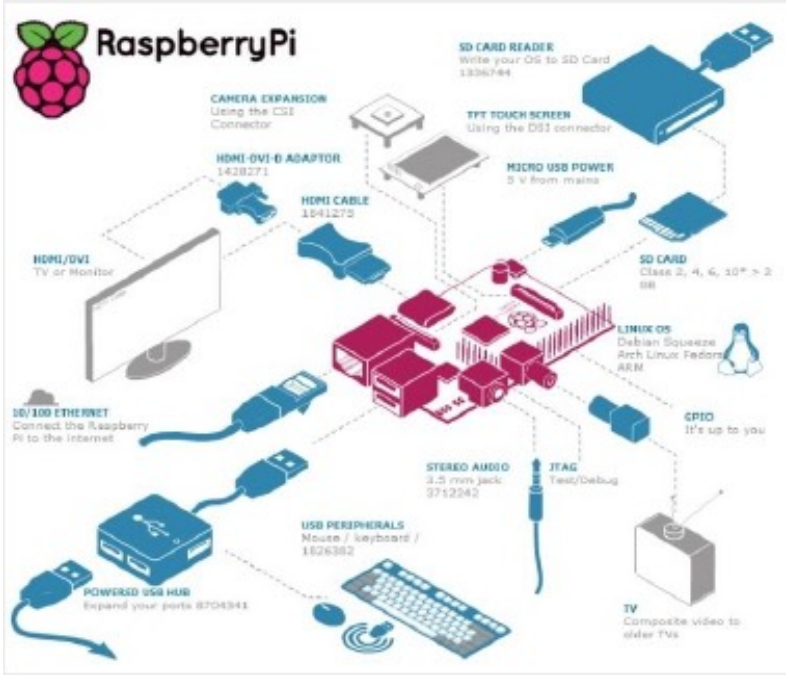
Things: HW alternatives for prototypes



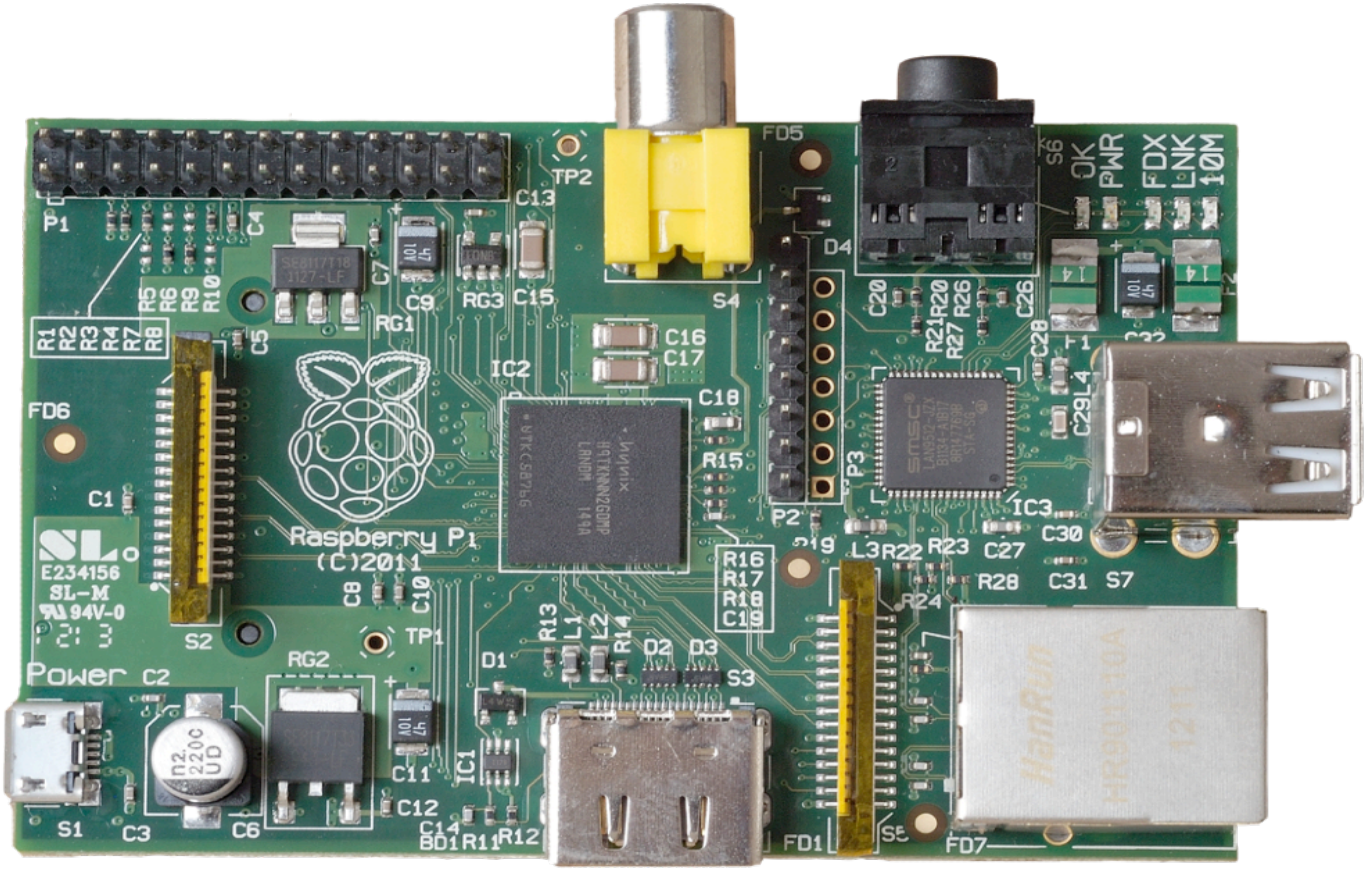
https://www.postscapes.com/internet-of-things-hardware/

Raspberry Pi: What is it?

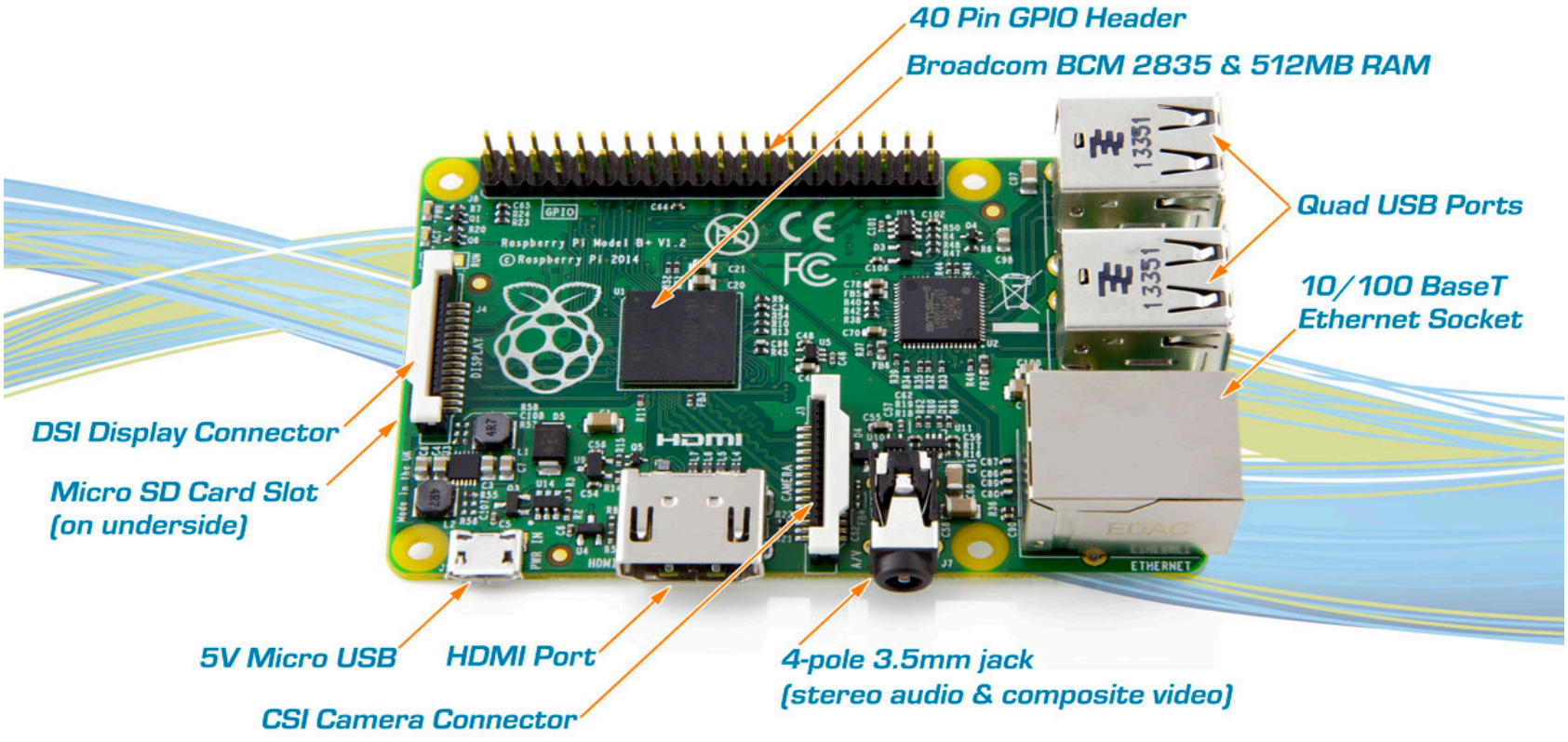
- A credit card sized single board computer
  - www.raspberrypi.org
- Other single board computers
  - https://en.wikipedia.org/wiki/Comparison_of_single-board_computers



The first Raspberry Pi



Raspberry Pi Model B+



Raspberry Pi 2 Model B

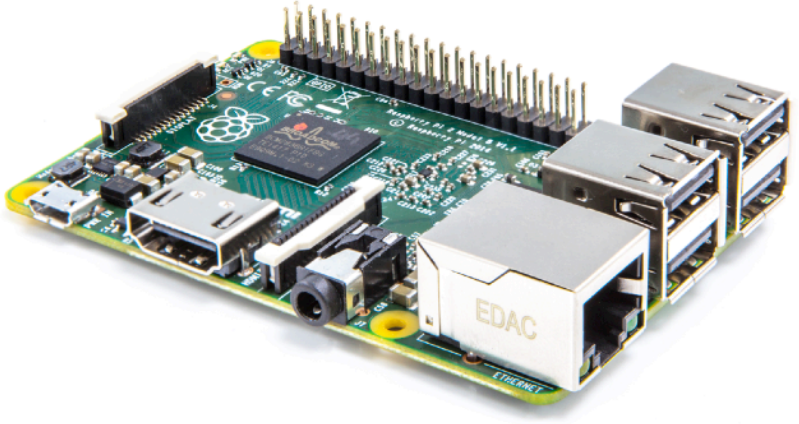
The Raspberry Pi 2 Model B is the second generation Raspberry Pi.
It replaced the original Raspberry Pi 1 Model B+ in February 2015.

Compared to the Raspberry Pi 1 it has:

- **A 900MHz quad-core ARM Cortex-A7 CPU**
- **1GB RAM**

Like the (Pi 1) Model B+, it also has:

- 4 USB ports
- 40 GPIO pins
- Full HDMI port
- Ethernet port
- Combined 3.5mm audio jack and composite video
- Camera interface (CSI)
- Display interface (DSI)
- Micro SD card slot
- VideoCore IV 3D graphics core

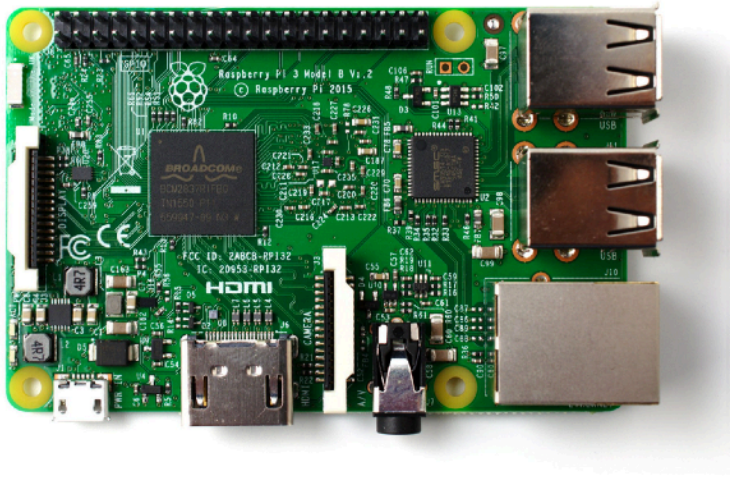


Because it has an ARMv7 processor, it can run the full range of ARM GNU/ Linux distributions, including Snappy Ubuntu Core, as well as Microsoft Windows 10.

The Raspberry Pi 2 has an identical form factor to the previous (Pi 1) Model B+ and has complete compatibility with Raspberry Pi 1.

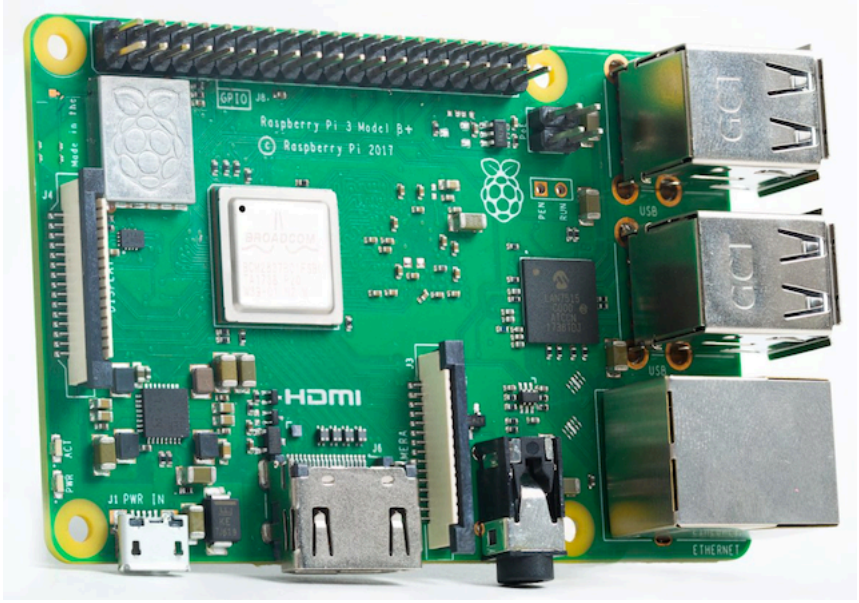
Raspberry Pi 3

- The Raspberry Pi 3 is the third generation Raspberry Pi. It replaced the Raspberry Pi 2 Model B in February 2016.
- Compared to the Raspberry Pi 2 it has:
  - A 1.2GHz 64-bit quad-core ARMv8 CPU
  - **802.11n Wireless LAN**
  - **Bluetooth 4.1 (BLE)**

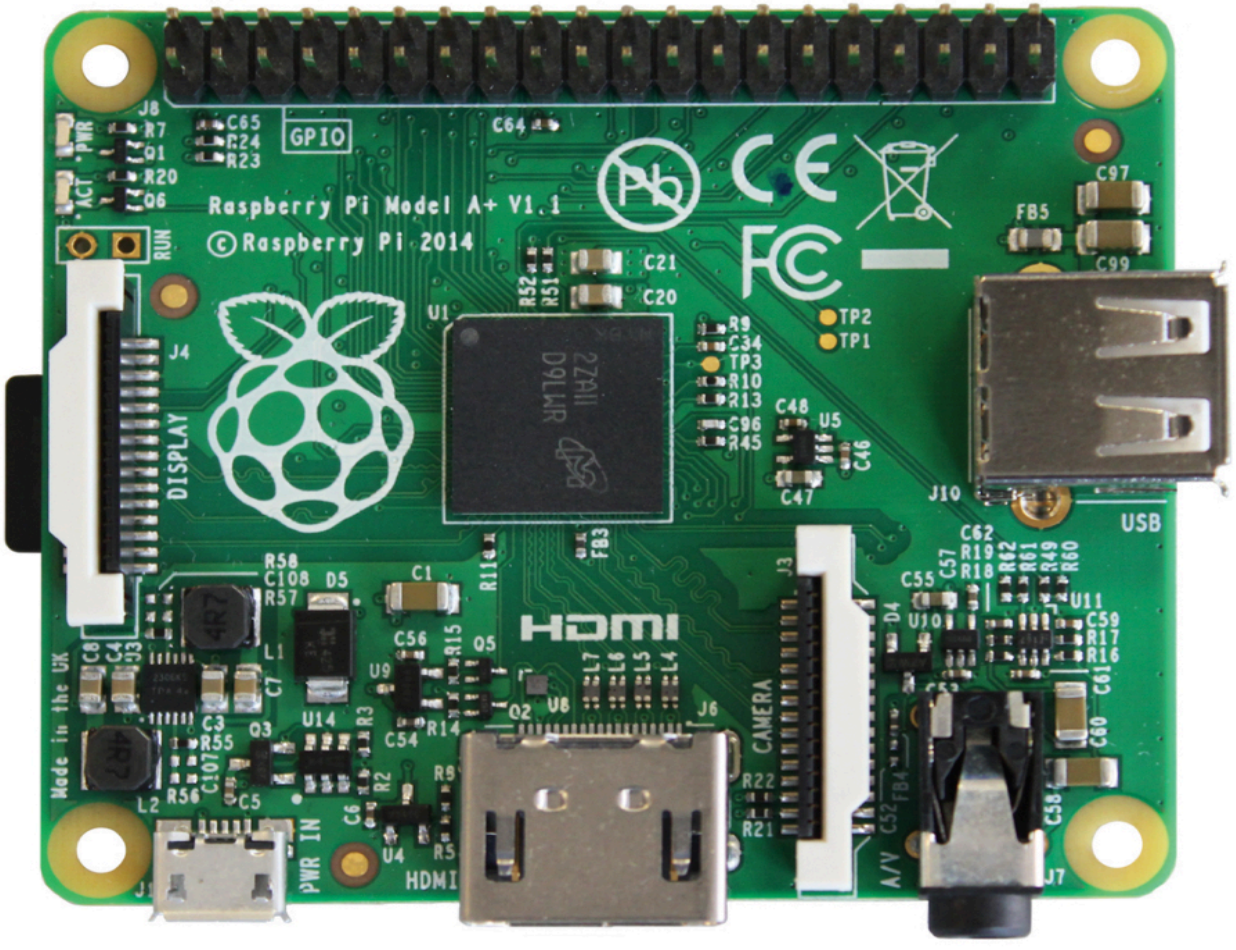


Raspberry PI 3 model B+

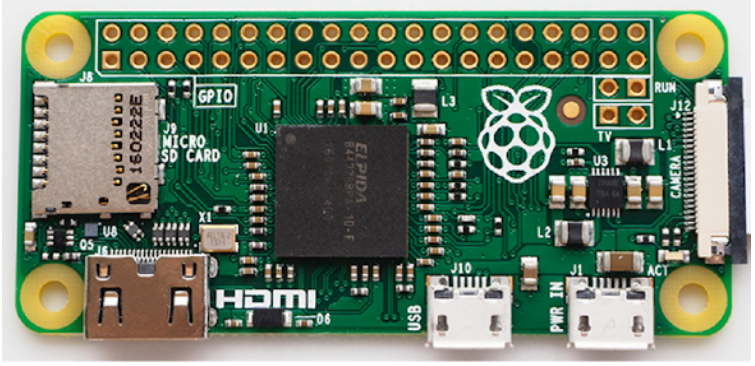
- Broadcom BCM2837B0, Cortex-A53 (ARMv8) 64-bit SoC @ 1.4GHz
- 1GB LPDDR2 SDRAM
- 2.4GHz and 5GHz IEEE 802.11.b/g/n/ac wireless LAN, Bluetooth 4.2, BLE
- Gigabit Ethernet over USB 2.0 (maximum throughput 300 Mbps)
- CSI camera port for connecting a Raspberry Pi camera
- DSI display port for connecting a Raspberry Pi touchscreen display
- Power-over-Ethernet (PoE) support (requires separate PoE HAT)

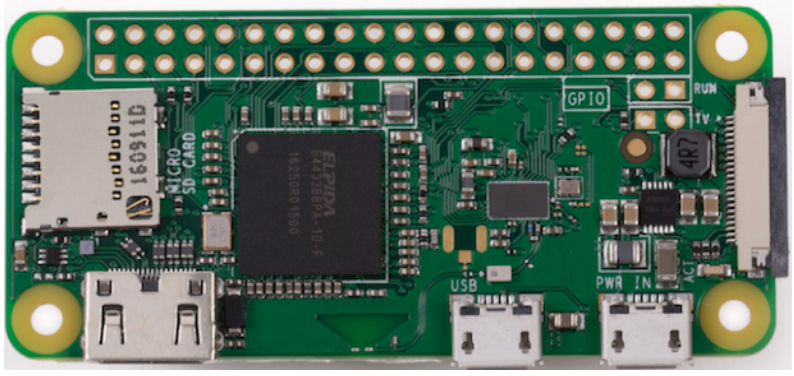


Raspberry Pi Model A+



Raspberry Pi Zero & Zero W





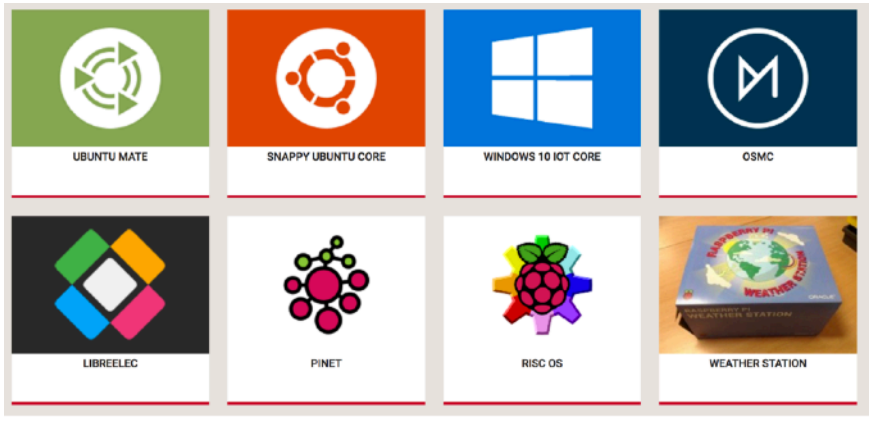
- 1GHz, single-core CPU
- 512MB RAM
- Mini HDMI and USB On-The-Go ports
- Micro USB power
- HAT-compatible 40-pin header
- Composite video and reset headers
- CSI camera connector

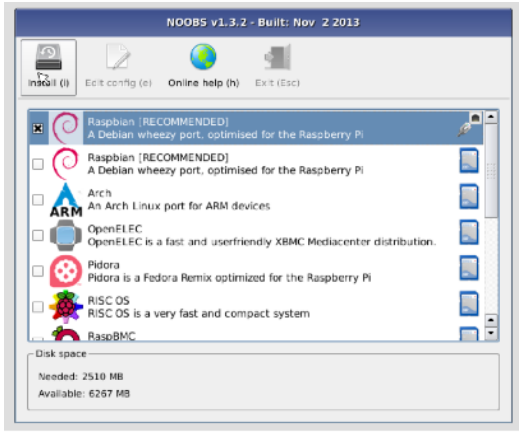
Plus:
- 802.11 b/g/n wireless LAN
- Bluetooth 4.1
- Bluetooth Low Energy (BLE)

Installing the OS

- Get the Raspian Image
  - http://www.raspberrypi.org/downloads

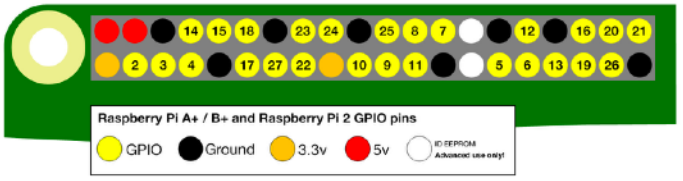
- Or get the NOOBS (New Out-Of-the Box Software)
- Install it on a SD card
  - https://www.raspberrypi.org/documentation/installation/



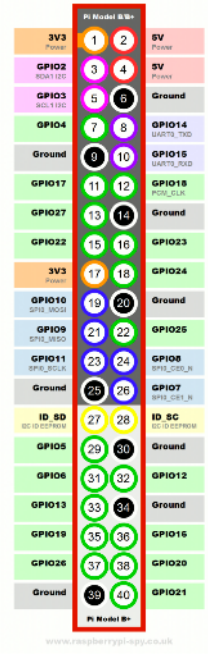


What is the GPIO?

- Stands for General Purpose Input/ Output
- https://www.raspberrypi.org/documentation/usage/gpio/
- It allows peripherals and expansion boards (such as the Gertboard) to access the CPU by exposing the inputs and outputs.
- C, C#, Java, Perl, Python, Ruby, Shell, Scratch language libraries included to read and write to the GPIO interface
- Examples on these pages
  - http://elinux.org/RPi_Low-level_peripherals
  - http://www.raspberrypi.org/archives/1471



GPIO Layout – Pi 3 Model B
Same layout as the :
•Pi Zero & Pi Zero W
•Pi 2 Model B
•Pi Model B+
•Pi Model A+

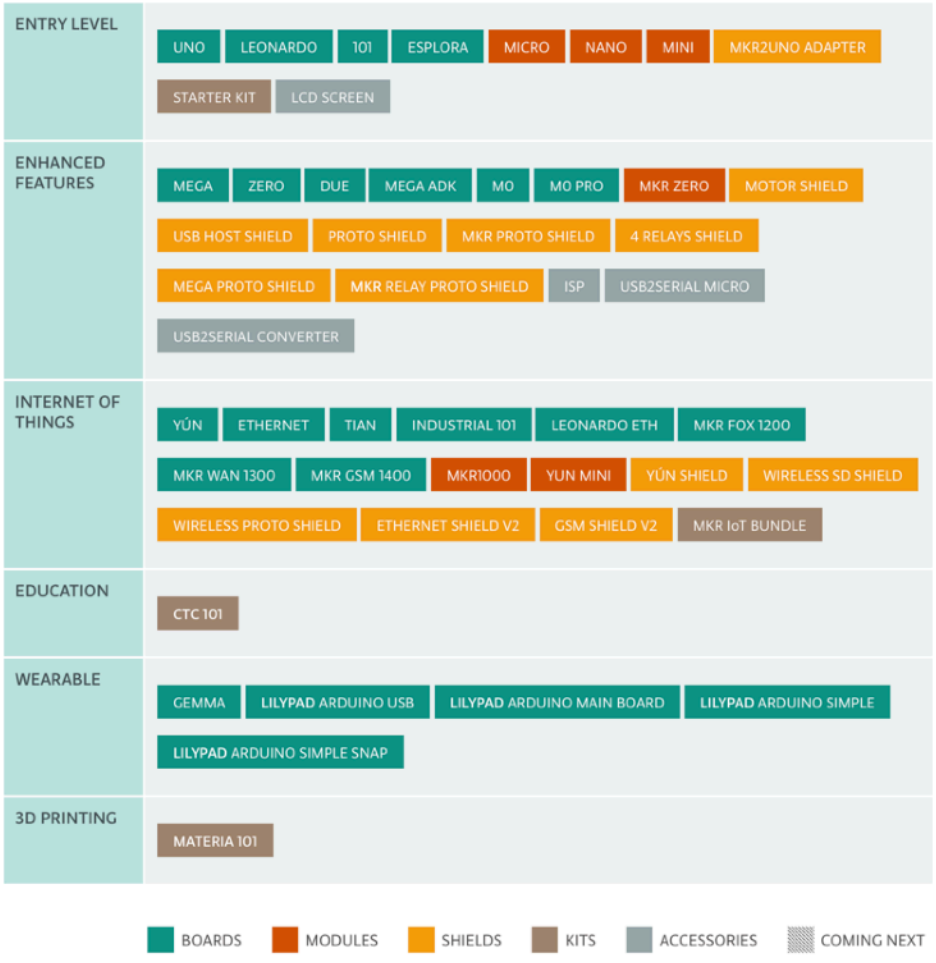


https://www.raspberrypi-spy.co.uk

Arduino

- Arduino is a widely used development board. This board is well known in the embedded community.
- Most Arduino boards are built using **Atmel AVR**, but some boards use other MCUs regarding to who joints venture with Arduino.
- Currently, Arduino boards are built by Arduino.cc.
  - Other companies also build boards, which are usually called Arduino-compatible. This is because the founder of Arduino already shared the board scheme so that people can build own Arduino.
- To extend Arduino I/O and functionalities, we can use Arduino shields.
  - There are many Arduino shields, with different purposes, for instance, Bluetooth, Wi-Fi, GSM, temperature, and humidity sensors.
  - We just have to attach Arduino shield to the Arduino board without any soldering.
- A comparison of all Arduino boards from Arduino.cc can be found here: https://www.arduino.cc/en/Products/Compare

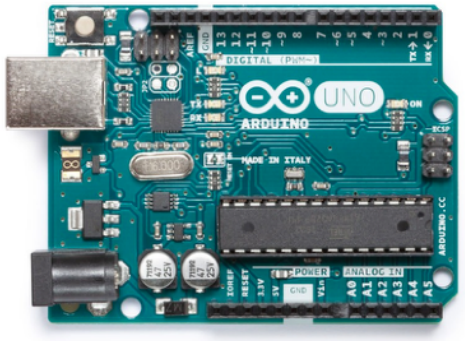
Arduino products

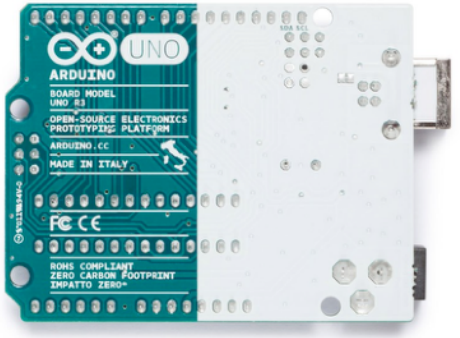


https://www.arduino.cc/en/Main/Products

Arduino Uno

- The Arduino Uno model is widely used in Arduino development.
- It's built on top of a MCU ATmega328P microcontroller
- The board provides several digital and analog I/O pins, to which we can attach our sensor and actuator devices.
- SPI and I2C protocols are also provided by the Arduino Uno.
- More infos: https://store.arduino.cc/arduino-uno-rev3

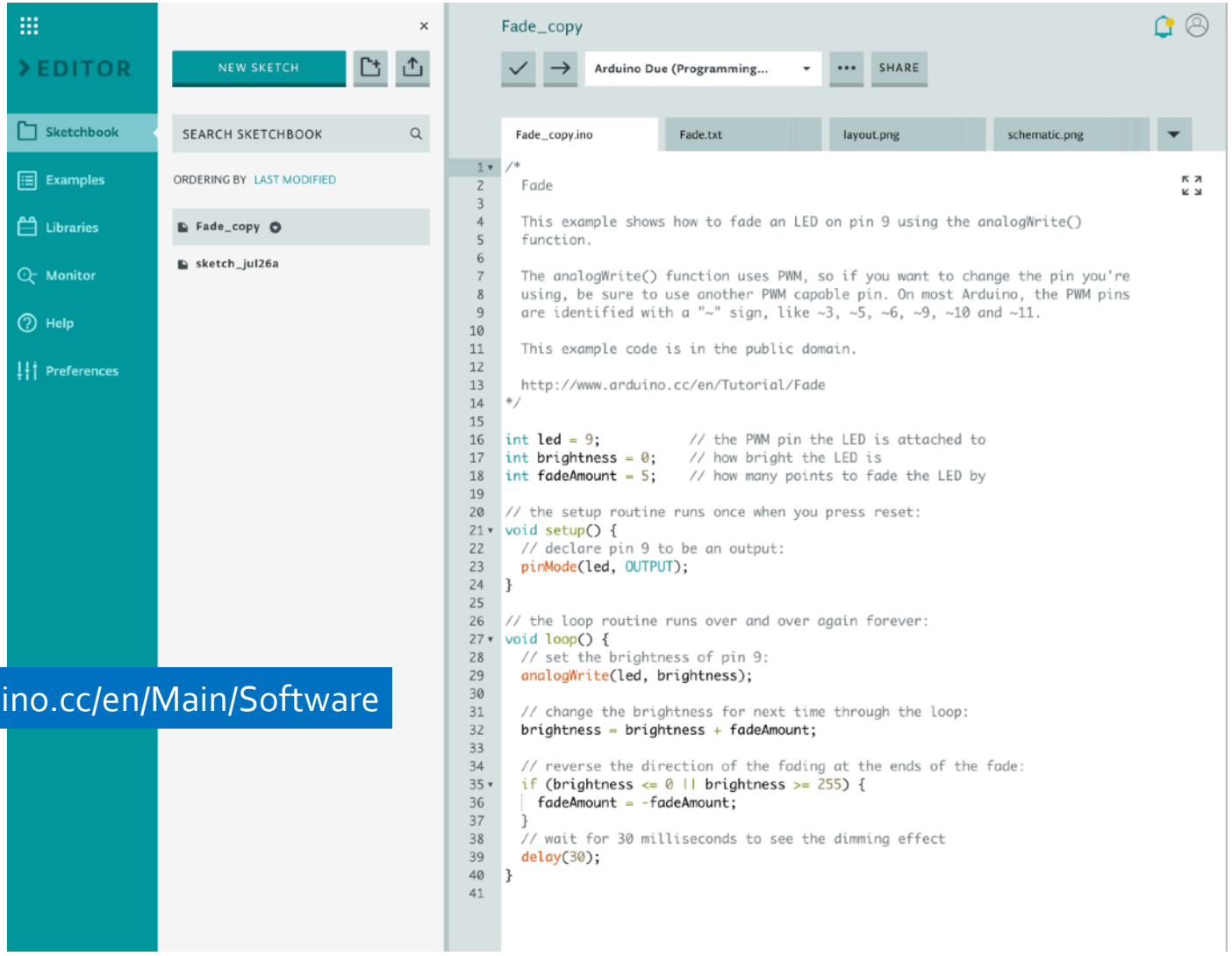




Arduino MKR1000

- Arduino MKR1000 uses the Atmel ATSAMW25 SoC, which provides a built-in Wi-Fi module.
- This is a nice IoT solution for the Arduino platform because the Wi-Fi module, WINC1500, is supported for SSL and ECC508 CryptoAuthentication.
- Further information: https://store.arduino.cc/arduino-mkr1000-with-headers-mounted

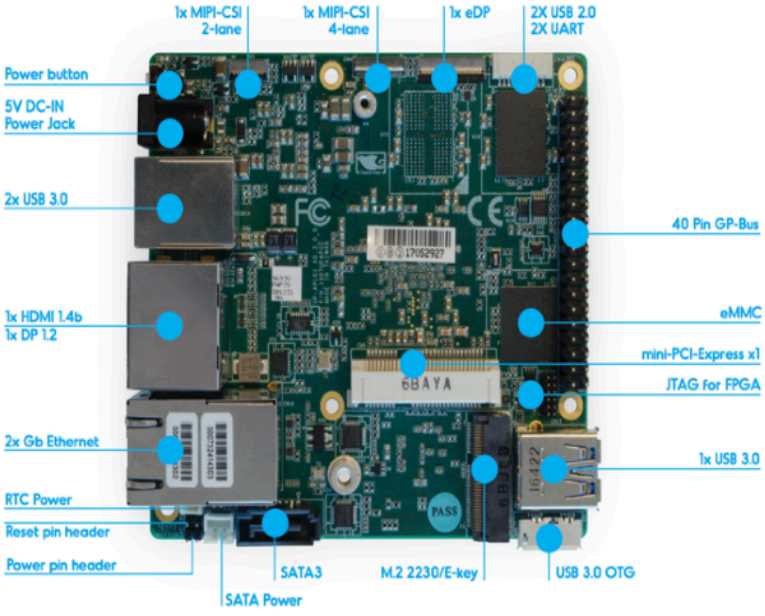
Arduino Web Editor



https://www.arduino.cc/en/Main/Software

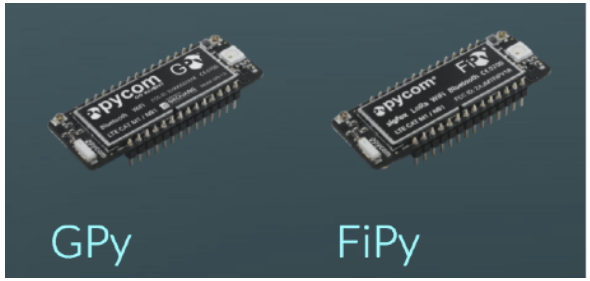
Nodes HW alternatives for prototypes: INTEL based

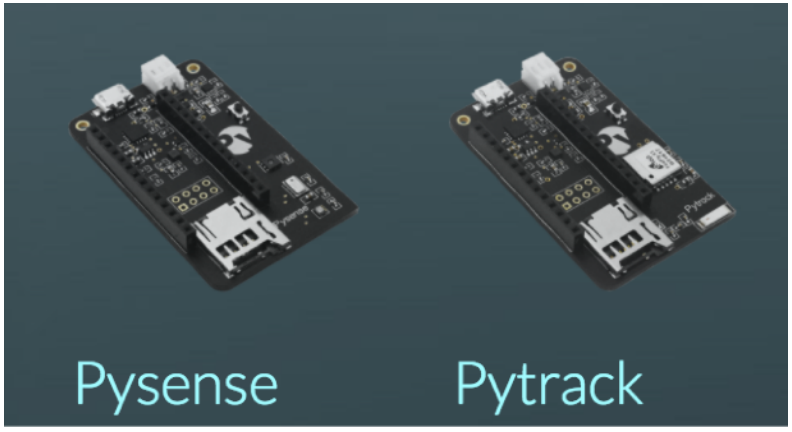
- UP² (Squared) is a x86 maker board based on Intel Apollo Lake platform
  - http://www.up-board.org/upsquared/
  - https://software.intel.com/en-us/iot/hardware/up-squared-grove-dev-kit



Nodes: HW alternatives for prototypes: Pycom

- Multi-network hardware.
  - Wifi, Bluetooth, Sigfox, LoRa and LTE-M
- Based on MicroPython
- The portfolio includes a comprehensive range of development boards, expansion boards, sensor shields and OEM hardware





https://pycom.io/hardware/

Nodes: HW alternatives for prototypes: Pycom

Pymate is a free widget-based Mobile App that let's you manage and control your Pycom board

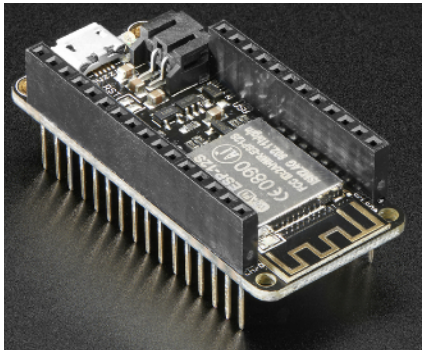
https://www.pycom.io/pymate/

Nodes: HW alternatives for prototypes: others

FIREFLY

RE-MOTE SUITE

http://zolertia.io/



Assembled Feather HUZZAH w/ ESP8266 WiFi With Stacking Headers

https://www.nxp.com

NXP Pico i.MX7D

NXP Pico i.MX6UL

NXP Argon i.MX6UL

NXP SprIoT i.MX6UL

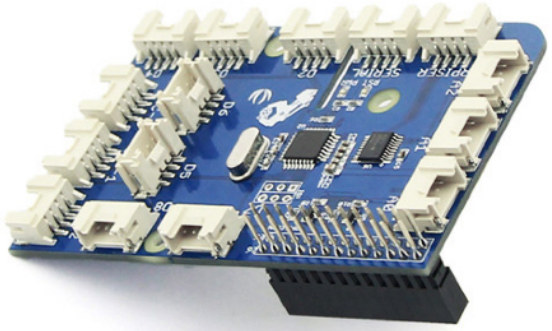
And much more here:
https://www.postscapes.com/internet-of-things-hardware/ or here
https://www.adafruit.com/category/342

Sensors HW

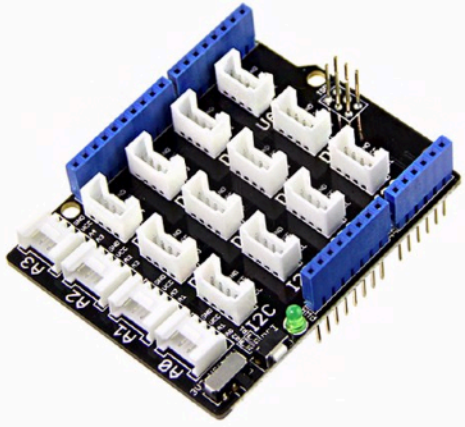
- There is plenty of specific sensors available in the market for many differente puposes
- They can be combined with the previously seen HW platform to obtain the specific combination of HW and SW to solve the specific problems we are interested into.
- In the following some relevant example are given

The Grove board

GrovePi+ is add-on board with 15 Grove 4-pin interfaces that brings Grove sensors to the Raspberry Pi. It is the newest version compatible with Raspberry Pi model B/B+/A+/ 2/3 perfectly.



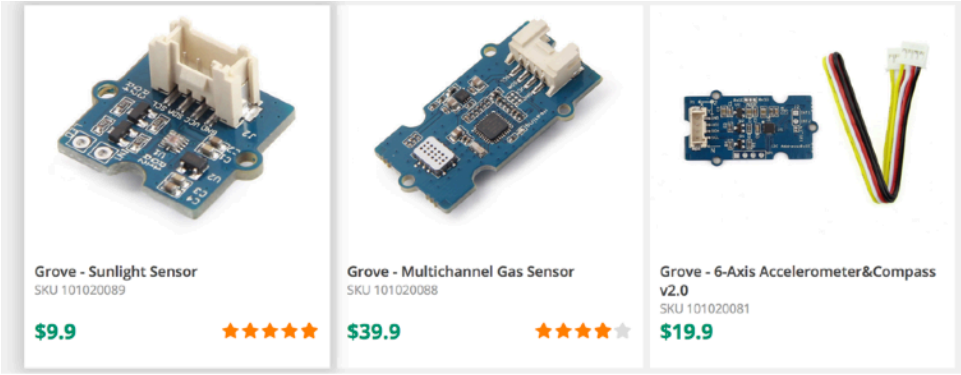
https://www.seeedstudio.com/GrovePi%2B-p-2241.html



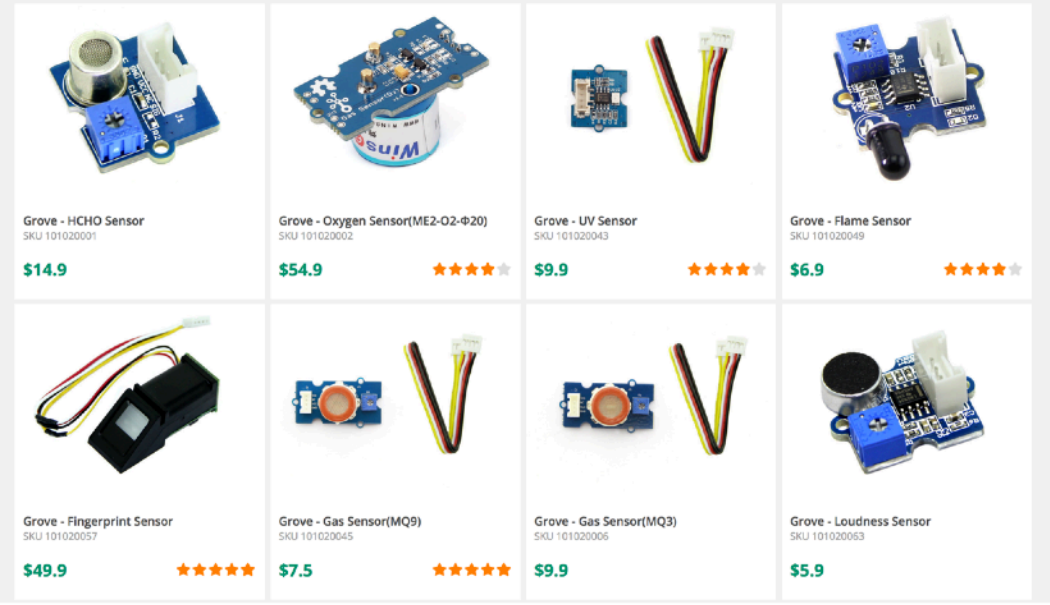
https://www.seeedstudio.com/Base-Shield-V2-p-1378.html

Grove – Base Shield is to help you getting rid of bread board and jump wires. Through the grove connectors on the base board, you can add all the grove modules to the Arduino Uno.

Sensors by seeedstudio

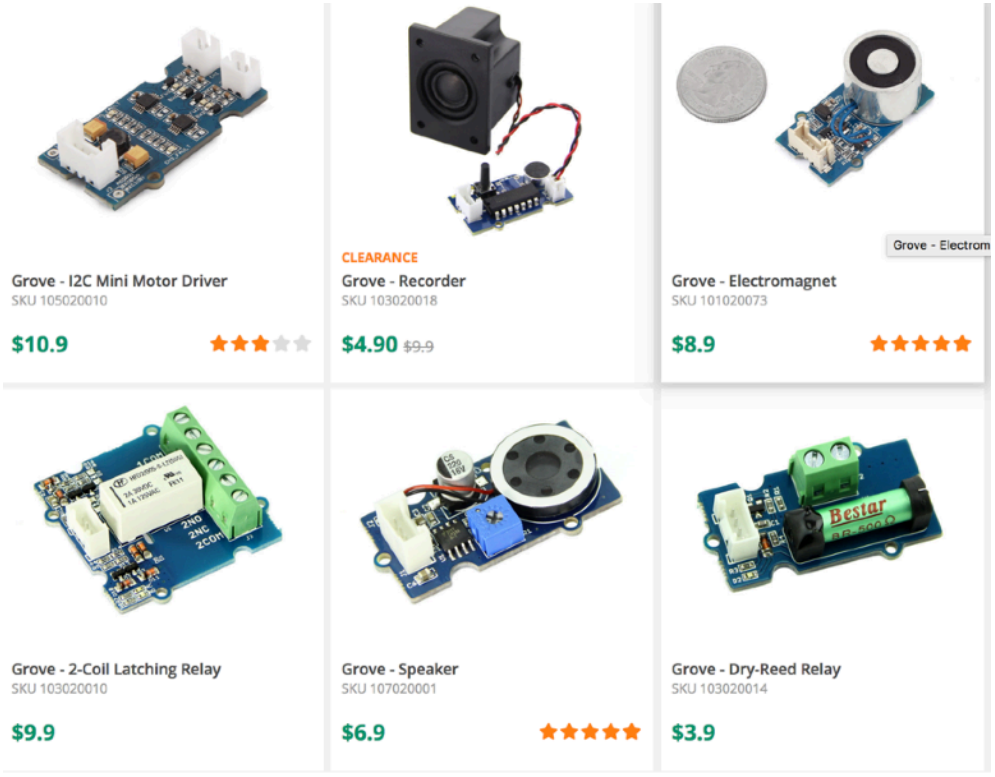


https://www.seeedstudio.com/category/Sensor-for-Grove-c-24.html



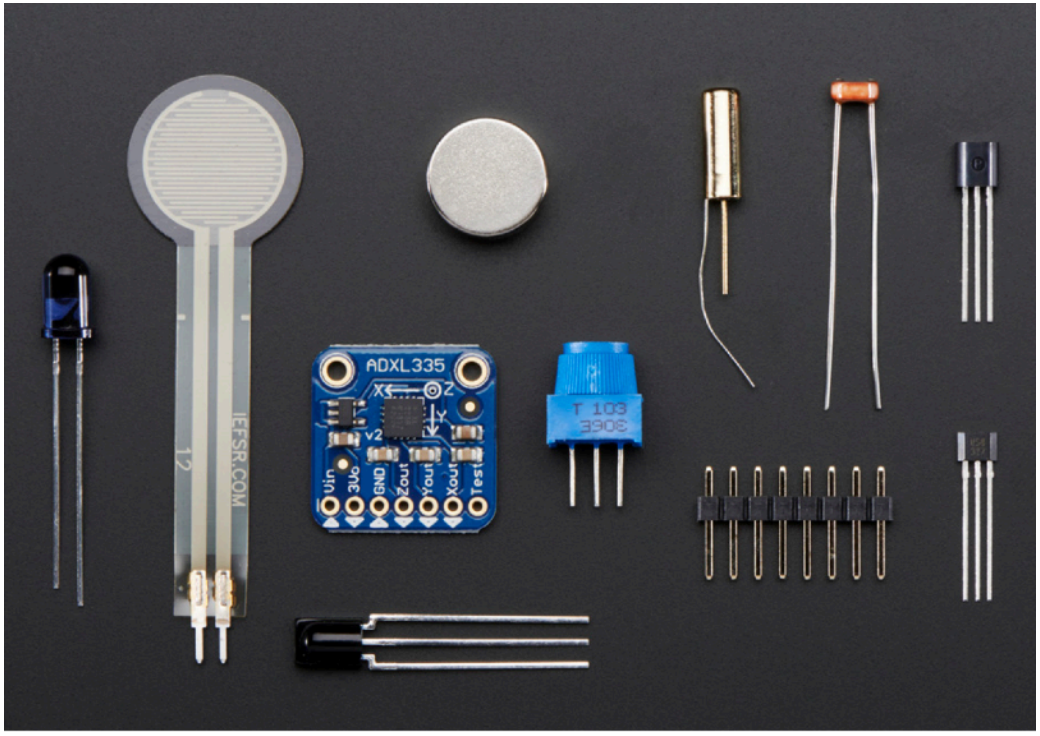
Actuators by seeedstudio

https://www.seeedstudio.com/category/Actuators-for-Grove-c-25.html



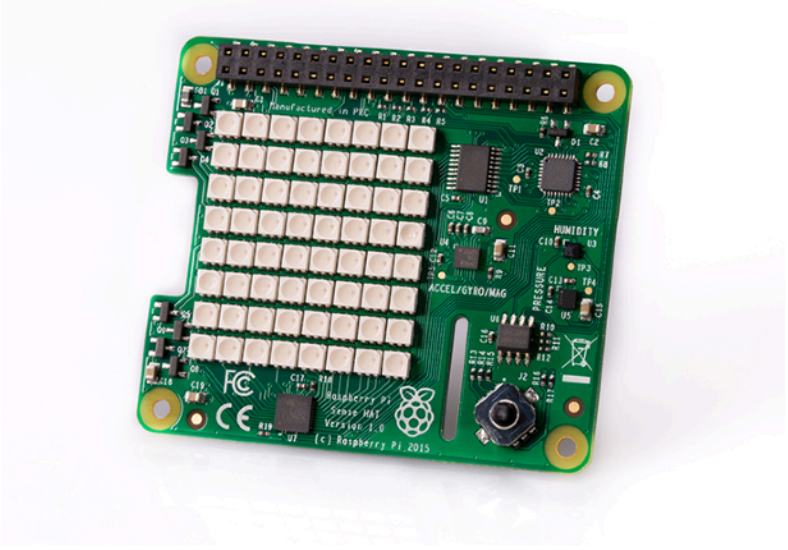
Sensors by Adafruit

- A lot of basic sensors can be found in https://www.adafruit.com/category/35



Sense Hat

- 8 x 8, 16-bit LED display that outputs sense data via shapes, icons and messages
- Inertial Measurement Unit (IMU) essential for velocity, orientation and gravity measurement of a space craft, the IMU combines an accelerometer, gyroscope and magnetometer into one chip
- Barometric air pressure sensor
- Temperature sensor
- Humidity sensor
- 5-button joystick that serves as an on-board keyboard and mouse
- Emulator: https://trinket.io/sense-hat



Some useful links

- IoT Sensors and Actuators
  - https://www.postscapes.com/trackers/video/the-internet-of-things-and-sensors-and-actuators/
- IoT hardware guide
  - https://www.postscapes.com/internet-of-things-hardware/
- http://iotlist.co/