IoTEC

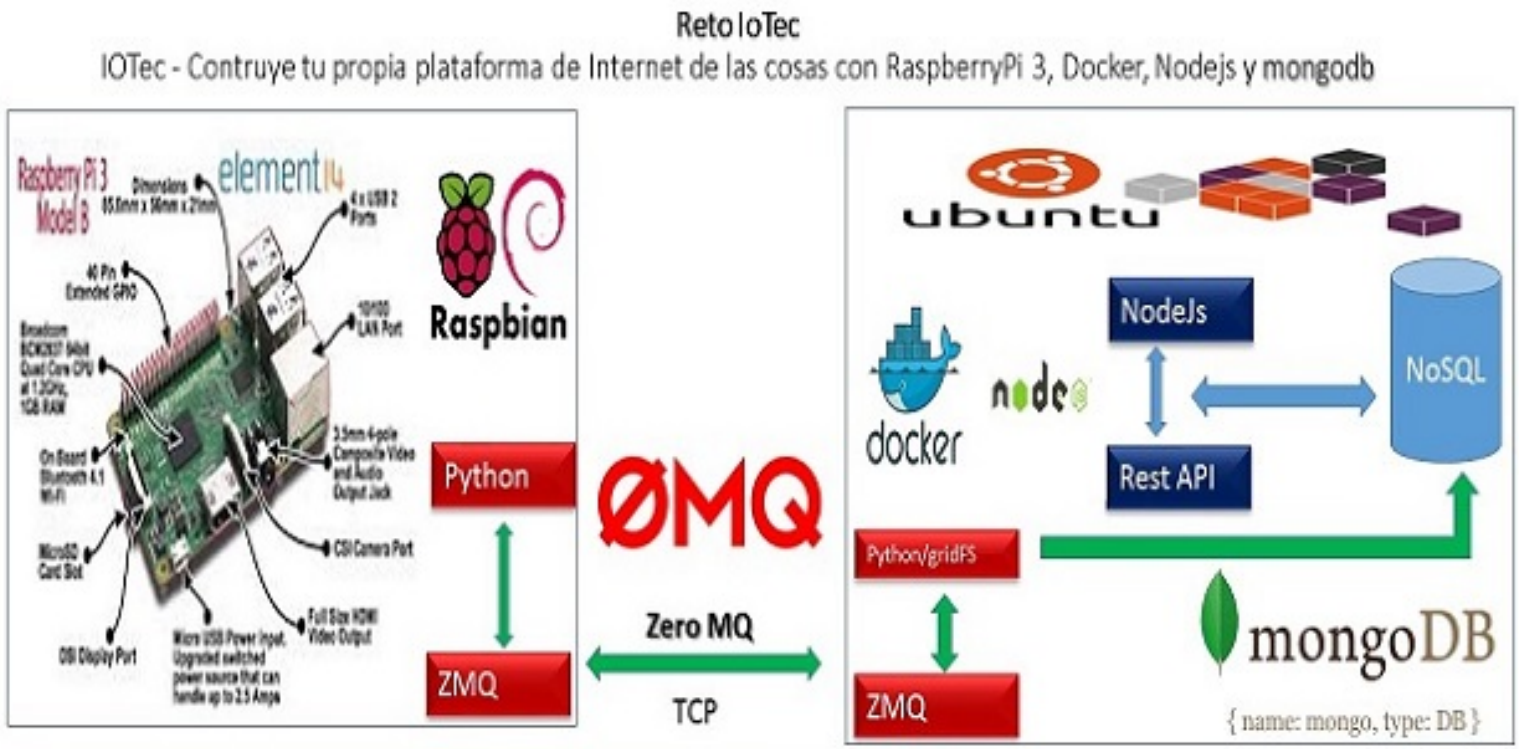
Adolfo Centeno, Daniel Perez, Fabio Galo

November 16, 2017

Table of contents

Internet of things

IoTEC - RoadMap



Ganaras las siguientes habilidades

1. Diseño de arquitecturas basadas en microservicios usando docker
2. Aprenderas base de datos NoSql como mongodb
3. Construiras y pondrás en producción una plataforma web con nodejs, rest-api con web-tokens, html5, boootstrap y jquery.
4. Instalaras y configuraras Linux raspbian sobre una raspberry pi 3
5. Programaras en Python y el framework zero-mq protocolos de comunicación para conectar la raspberry pi con el servidor en la nube
6. Aprenderas como persistir los datos e imágenes transmitidas en tiempo real con Python y gridFS

IOTEC - Schedule

- Day 1 (Introduction, Google Compute Engine Client and Mongodb database)
- Day 2 (Docker, Nodejs)
- Day 3 (Raspbian install, python, zero-mq)
- Day 4 (Programming sockets with python/zero-mq)
- Day 5 (Final project)

The cloud computing stack – SaaS, PaaS, and IaaS

We can represent cloud computing as a stack of three different categories:

- Software as a Service (SaaS)
- Platform as a Service (PaaS)
- Infrastructure as a Service (IaaS)

Hosting + Compute

There are two options if we want to host an application on Google Cloud Platform:

1. Google App Engine: This is Google's PaaS and it will not be covered in this project.
2. Google Compute Engine: This is Google's IaaS and lets users run virtual machines on Google's infrastructure with a variety of hardware and software configurations.

Ubuntu/Debian install (Part I)

Create an environment variable for the correct distribution

```
export CLOUD_SDK_REPO="cloud-sdk-$(lsb_release -c -s)"
```

Add the Cloud SDK distribution URI as a package source:

```
echo "deb http://packages.cloud.google.com/apt $CLOUD_SDK_REPO main" |
    sudo tee -a /etc/apt/sources.list.d/google-cloud-sdk.list
```

Import the Google Cloud public key:

```
curl https://packages.cloud.google.com/apt/doc/apt-key.gpg | sudo apt-key add -
```

Ubuntu/Debian install (Part II)

Update and install the Cloud SDK:

```
sudo apt-get update && sudo apt-get install google-cloud-sdk
```

Install the additional component:

```
sudo apt-get install google-cloud-sdk-app-engine-python
```

Ubuntu/Create new user

add new user

```
$ sudo adduser [username]              # add user
$ sudo usermod -aG sudo [username]     # grant root privileges
$ su [username]                        # swith user to [username]
$ whoami                               # display current user
$ cd ~                                 # change directory to user's home
$ pwd                                  # display path working directory
```

Run gcloud auth login to get started:

```
$ sudo gcloud auth login  # login to Google
```

NOTE:
gmail user : mindlesssmile72@gmail.com
password : semanai2017

Connect to Google Compute Engine

Set project/connect with ssh to GCE

```
# set the default project
$ sudo gcloud config set project adsoft-iosclient

# print virtual private servers
$ sudo gcloud compute instances list

# check network connectivity
$ ping 35.185.213.109

# login with secure shell (ssh) to compute instance with [username]
$ sudo gcloud compute ssh [username]@mindlesssmile72 --zone us-west1-b

# show home directory
$ ls /home
```

MongoDB - Overview

What's MongoDB

- MongoDB is a document database with the scalability and flexibility
- MongoDB stores data in flexible, JSON-like documents
- MongoDB is free and open-source
- TCP port: 27017
- MongoDB connection: mongodb://35.185.213.109:27017/ourdatabase

Start/Stop/Status/Restart mongo

```
# start mongodb
$ sudo service mongod start

# start mongodb
$ sudo service mongod status

# connect to mongo
$ mongo localhost   # other example:  mongo 35.185.213.109:27017

$ show databases;

# exit from mongo shell
$ exit
```

Create database

```
# show databases
> show dbs

# create database
> user iotec -[username]

# create device collection
> db.devices.insert({"name":"edison"})

$ show collections;
> show collections

# show devices
$ db.devices.find()
```

Create

Create documents

```
# create one
> db.devices.insertOne({"name":"lego mindstorm ev3"})

# create many
> db.devices.insertMany([{"name":"arduino one"}, {"name": "arduino mega"}])

# list all devices
> db.devices.find()
```

Read

Query documents

```
# query filter
> db.devices.find({"name":"edison"})

# contains query filter
> db.devices.find({name: /^arduino/})

# list top n devices
> db.devices.find().limit(3)

# in
> db.devices.find( { name : { $in: [ "edison", "arduino one" ] } } )


# and
> db.devices.find({_id: ObjectId("59c8975c95b0585b53e4b728")}, {name: /^arduino/})

# or
> db.devices.find({$or: [{name: "intel edison"}, {name: "dron pxfmini"}]});
```

Update

Update documents

```
# update one document
> db.devices.updateOne({"name":"intel edison"}, {$set: {name: "edison"}})
> db.devices.find()

# update many documents
> db.devices.updateMany({"name": /^arduino/}, {$set: {name: "arduino"}})
> db.devices.find()


# update one document using update operator (update the first document)
> db.devices.update({"name":"arduino"}, {$set: {name: "arduino one"}})
> db.devices.find()

# update many documents, using update operator (multiples update)
> db.devices.update({"name": /^arduino/}, {$set: {name: "arduino"}}, {multi : true})
> db.devices.find()
```

Delete

Delete documents

```
# delete one document
> db.devices.deleteOne({"name":"edison"})
> db.devices.find()

# delete many documents
> db.devices.deleteMany({"name": /^arduino/})
> db.devices.find()

# delete one document wirh remove operator
> db.devices.remove({"name": /^lego/}, 1)
> db.devices.find()

# delete many documents with remove operator
> db.devices.remove({"name": /^dron/})
> db.devices.find()
```