

Performance Evaluation for IoT

Test complex scenarios

Andrea Vitaletti



Note: the images are linked to the sources

The cloud

- Big data processing
- Data warehouses



The edge

- Real time data processing
- Local processing



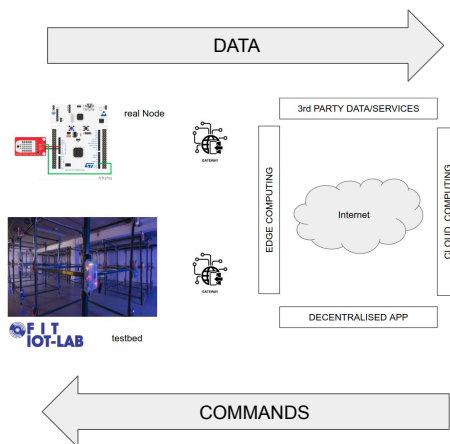
Internet of things

- Smart devices
- Smart vehicles
- Connected systems



Edge computing

Edge computing allows data from internet of things devices to be analysed at the edge of the network before being sent to a data centre or cloud.



Your final deployment, in its essence, should look similar to the one depicted in the picture.

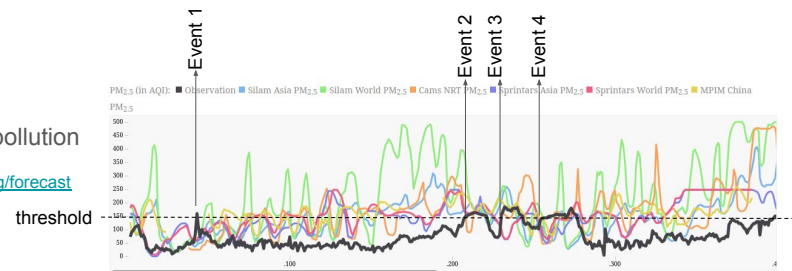
- The “real node” is used to prove your ability of integrating the specific sensors envisioned in your application scenario
- The lot-lab deployment is used to prove you ability to scale-up your solution into an ecosystem of nodes organised in a network and to evaluate their performance

What kind of phenomena?

- Sampling
 - reconstruct a signal (e.g. black PM2.5)
 - observe a value ... you already know the dynamics!
- Events (e.g. >threshold)

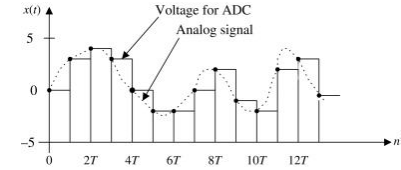
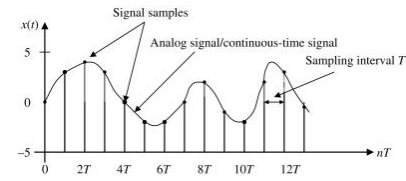
PM2.5 air pollution

<https://aqicn.org/forecast>



Reconstruct a signal

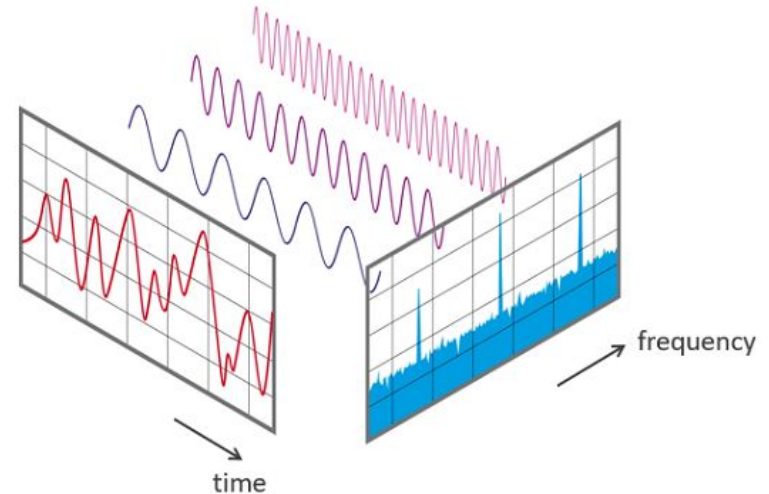
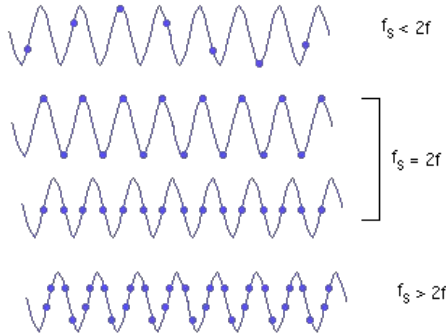
Sampling Theorem in few slides

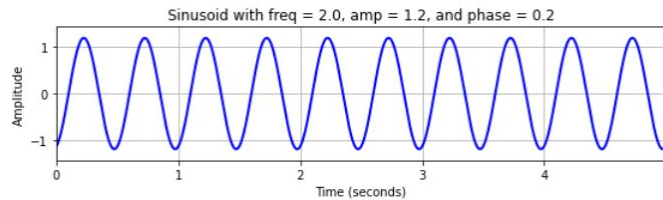


<https://www.sciencedirect.com/topics/computer-science/shannon-sampling-theorem>

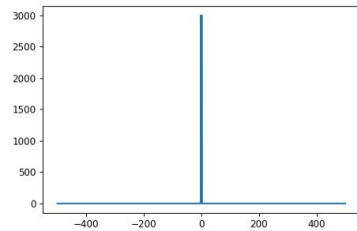
The **sampling theorem** specifies the minimum-**sampling** rate at which a continuous-time signal needs to be uniformly **sampl**ed so that the original signal can be completely recovered or reconstructed by these samples alone.

$$f_s > 2f_{max}$$

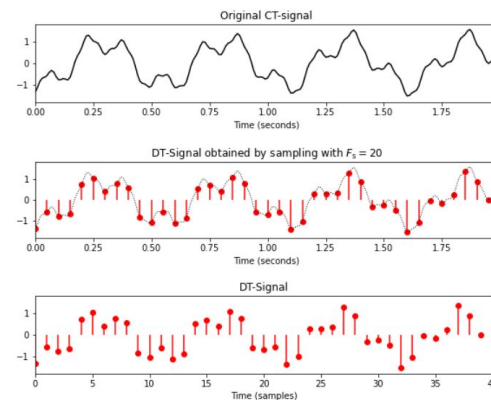
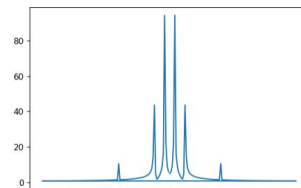




$$x = \sin(2\pi \text{freq} \cdot t - \text{phase})$$

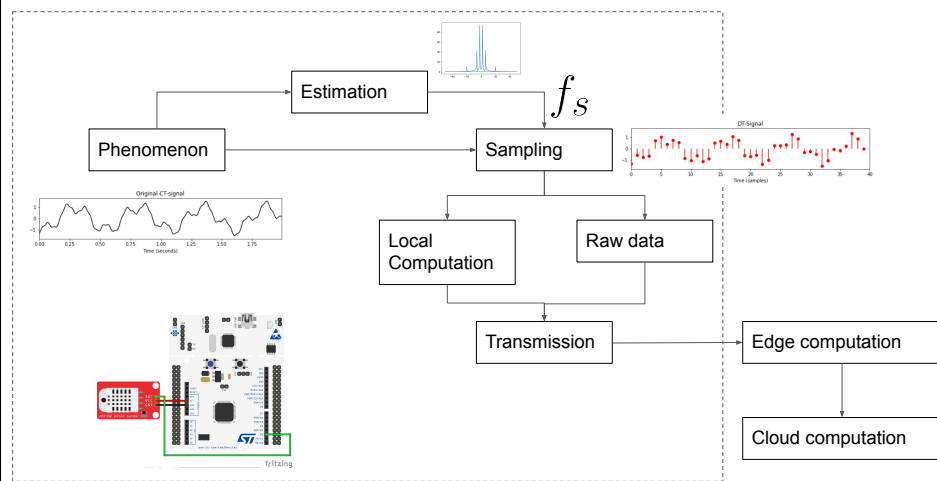
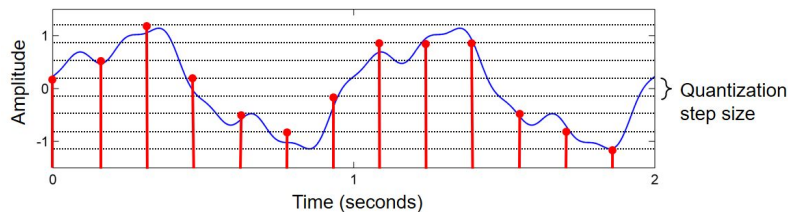
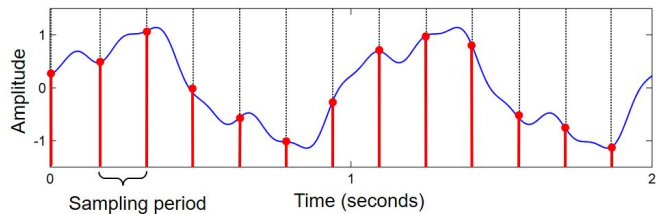


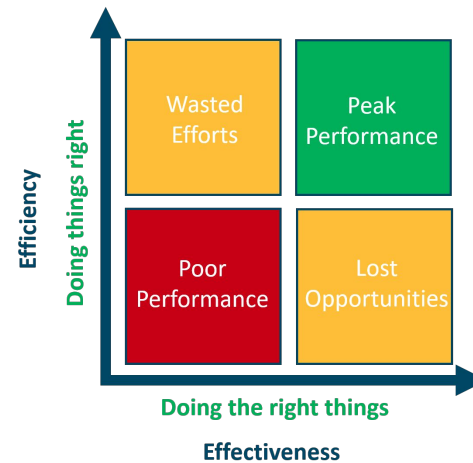
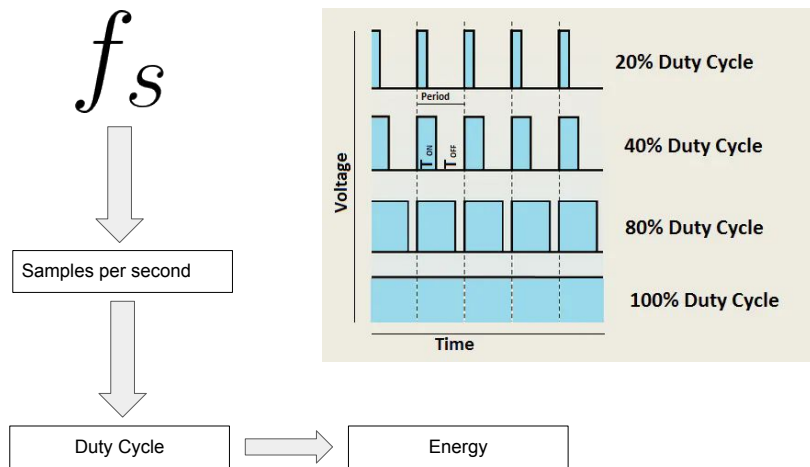
$$x = \sin(2\pi \cdot 1.9t - 0.3) + 0.5\sin(2\pi \cdot 6.1t - 0.1) + 0.1\sin(2\pi \cdot 20t - 0.2)$$



jupyter

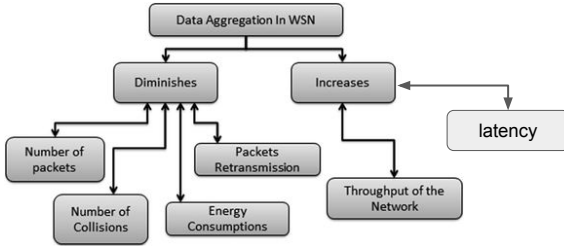
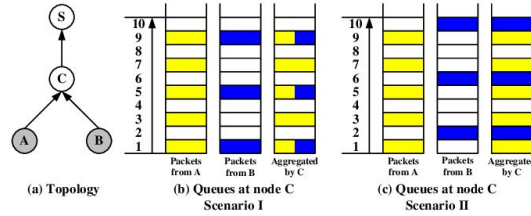
Figure 2.13 from [Müller, FMP, Springer 2015]





<u>NETWORK LAYER</u>	<u>QoS STRATEGY</u>	<u>IMPROVEMENT in QoS METRIC</u>
Application Layer	<ul style="list-style-type: none"> Compression Adjust Sensing Rate Data Reduction Techniques 	<ul style="list-style-type: none"> Latency, Available Bandwidth Network Lifetime, QoS Latency, Available Bandwidth
Network Layer	<ul style="list-style-type: none"> Routing Protocols Packets Priority 	<ul style="list-style-type: none"> Reliability, Latency, Network Lifetime Reliability, Latency, Network Lifetime
Physical Layer	<ul style="list-style-type: none"> Energy Aware MACs Selection of low interference channels MAC that Avoid Collisions 	<ul style="list-style-type: none"> Network Lifetime Reliability, Network Lifetime Reliability, Network Lifetime
Link Layer	<ul style="list-style-type: none"> Channel Surfing Modifying Signal Power Using low interference channel 	<ul style="list-style-type: none"> Reliability, Network Lifetime Network Lifetime Reliability, Network Lifetime

Aggregation



- ### IoT/Mist Metrics:
- Resource Utilization (Container)
 - Resource Load (Container)
 - IoT device Lifetime
 - Maximum running containers
 - Response Time (millisecond)
 - Delay/latency Time (ultra-low)
 - Provisioned Containers
 - Deprovisioned Containers
 - The number of Damaged Tasks (highly possible)
 - Energy Consumption (IoT devices)
 - Temperature (IoT device)
 - Reliability (critical)
 - Availability and Trustworthiness
 - Security (controversial)

- ### Edge Metrics:
- Resource Utilization (Container or VM)
 - Resource Load (Container)
 - Edge device lifetime
 - Maximum running container or VM
 - Response time (millisecond)
 - Delay/latency time (very low)
 - Provisioned Containers or VMs
 - Deprovisioned Containers or VMs
 - The number of Damaged Tasks (possible)
 - Energy Consumption (e.g., gateway)
 - Temperature (e.g., gateway or routers)
 - Reliability (moderate)
 - Availability and Trustworthiness
 - Security

- ### Common Metrics:
- Throughput
 - Network Congestion/Traffic Control
 - SLA Violation (i.e., success rate, dropped tasks)
 - Fault Tolerance
 - Statistical Analysis Measurements
 - The Number of Orchestration Decisions (e.g., in scheduling)
 - The number of Contradictory Decisions
 - Time to Adaptation/Scalability
 - Competition Ratio
 - Cache Hit Ratio
 - The number of Contradictory Actions (e.g., in placement)
 - Oscillation Mitigation (e.g., in scaling)
 - Cost/Profit
 - Technique's Overhead/Lightness
 - Privacy

- ### Fog Metrics:
- Resource Utilization (Container or VM)
 - Resource Load (Container or VM)
 - Resource lifetime (Server, Container, or VM)
 - Maximum running container or VM
 - Response time (millisecond or second)
 - Delay/latency time (very low)
 - Provisioned Containers or VMs
 - Deprovisioned Containers or VMs
 - Energy Consumption (e.g., fog servers)
 - Temperature (e.g., fog servers)
 - Reliability (moderate)
 - Security

- ### Cloud Metrics:
- Resource Utilization (Container, VM, host or data center)
 - Resource Load (Container, VM, host or data center)
 - Resource lifetime (Data center, Host, VM or Container)
 - Maximum Running Container, VM, host or data center
 - Response time (second)
 - Delay/latency time (low)
 - Provisioned Containers, VMs, Hosts or Data centers
 - Deprovisioned Containers, VMs, Hosts or Data centers
 - Energy Consumption (e.g., hosts and data centers)
 - Temperature (e.g., data centers)

https://www.researchgate.net/figure/A-taxonomy-of-real-time-performance-metrics-for-evaluating-LoT-Mist-Edge-Fog-and-Cloud_fig2_343578510

iot-lab.info/legacy/tutorials/index.html

(42) WhatsApp Imported Scimago Jour... didattica Trello



NEWS PLATFORM DEV CENTER COMMUNITY GET STARTED

Nodes Monitoring

Consumption



Radio

Monitor M3 consumption

Monitor consumption during experiment for a M3 node.



Radio monitoring for M3 nodes

Monitor radio activity during experiment for a M3 node.



Radio sniffing with M3 nodes

Capture and analyze radio communication during an experiment.

github.com/RIOT-OS/RIOT/tree/master/tests/periph_pm

Apps Reti di Calcola... (42) WhatsApp Imported



Home Docs Learn Community Blog

Getting started

Boards

Deployment

OS

Tools

API

CLI

SSH CLI

Run Script

Consumption monitoring

Radio monitoring

Websocket client

Serial aggregator

On-chip debugging

Radio characterization

MQTT broker

Leshan broker

Consumption monitoring

Consumption monitoring is an optional feature which measures the energy usage of your experiment nodes. It refers to the Control Node dedicated hardware installed on the IoT-LAB node to enable the monitoring. It provides you an efficient passive monitoring solution which helps you to design IoT protocols or applications with low-power devices. In this documentation you will learn how to create a Profile monitoring configuration and enable it for your experiment. Moreover you will figure out how to get and analyse the monitoring data.

Create a monitoring profile

You have two ways to create a monitoring *Profile*, the IoT-LAB Webportal or CLI command-line tools. In both cases you must create a Profile with a name, M3 architecture and the following configuration:

- Monitor consumption: current, voltage and power.
- Period: 8244 μ s
- Average: 4

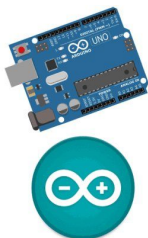
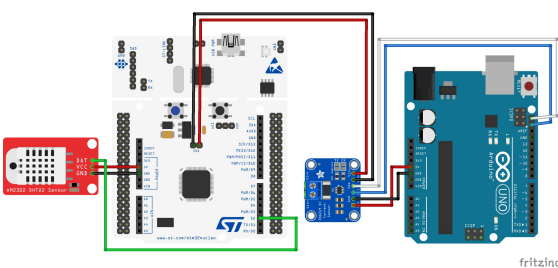
These settings will give you a sampling period of $P = 8.244 \text{ ms} \cdot 4 \cdot 2 = 65.95 \text{ ms}$. View this section for [additional informations](#).

In the Webportal go to the [Resources Monitoring](#) page and click [new profile](#) button.

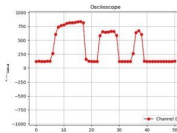
With command-line tools use these commands:

```
$ iotlab-auth -u <login> # optionally store your credentials if you haven't done it before.
$ iotlab-profile addm3 -n <profile_name> -p -voltage -current -power -period 8244 -avg 4
```

https://github.com/RIOT-OS/RIOT/tree/master/tests/periph_pm

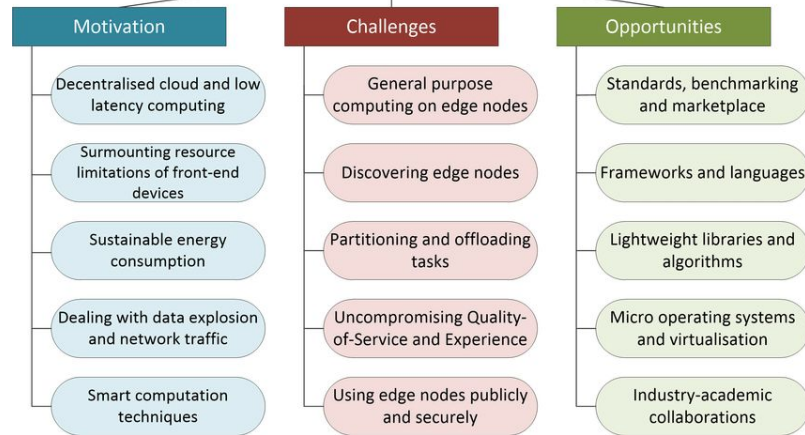


Arduino Oscilloscope



https://github.com/ichatz/riotos-apps/tree/main/temperature_humidity

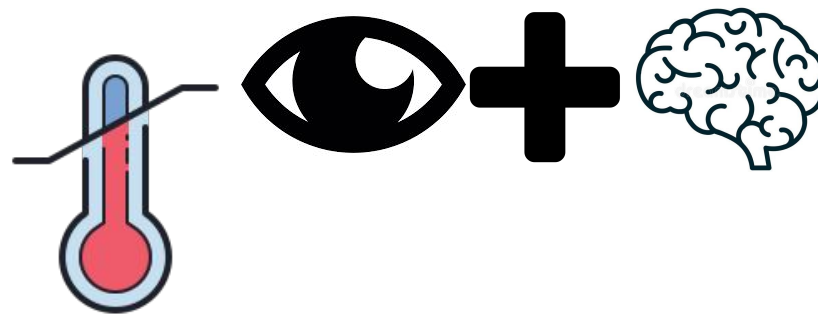
EDGE COMPUTING



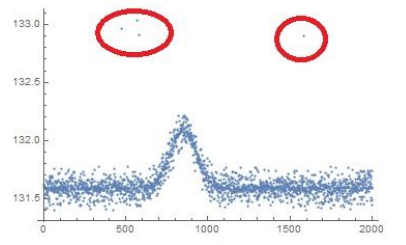
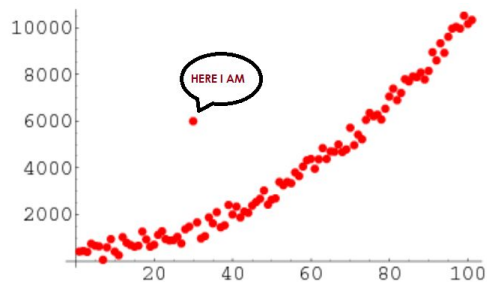
More on Lecture 14

Events

Something interesting happens!



Outliers: interesting events or noise?



A short journey of outlier detection

Quick overview of several methods for finding outliers



Ryota Bannai Jan 12, 2019 · 5 min read



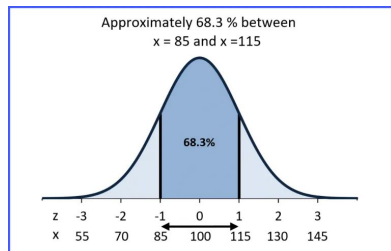
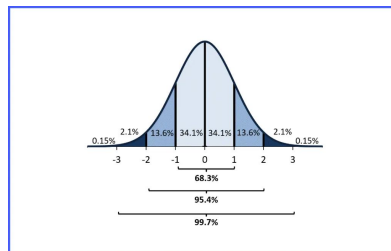
Let's focus on Z-score

$$z = \frac{x - \mu}{\rho}$$

where

- x is raw data
- μ is just mean of x
- ρ is standard deviation of x .

OUTLIER: x whose $|Z\text{-Score}| > \text{threshold}$



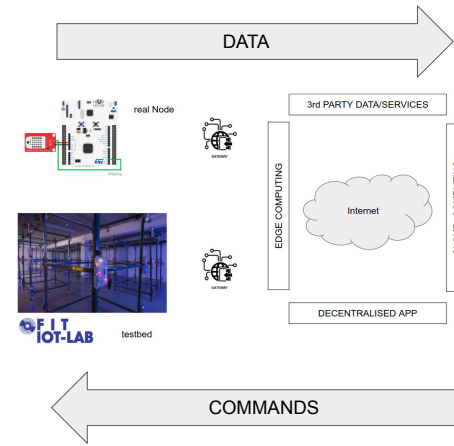
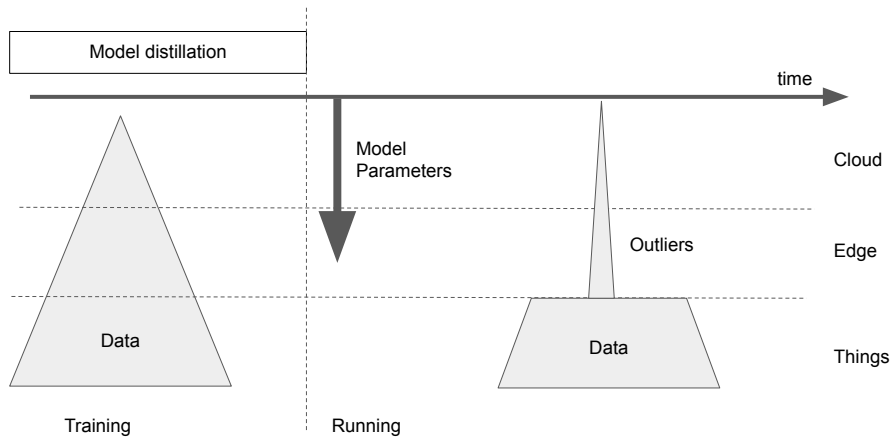
Source: <https://www.drdownright.com/empirical-rule-and-z-score-probability/>

```
1 import numpy as np
2 x=np.array([1,4,7,2,5,7,7,8,4,6,8,30])
3 z=(x-x.mean(axis=0))/x.std()
4 #
5 for i in range(len(z)):
6     if z[i] > 3 or z[i] < -3:
7         print(z[i])
```

outlier_w_zscore.py hosted with ❤ by GitHub [view raw](#)

$$z = \frac{x - \mu}{\rho}$$

Where are we going to compute μ and ρ ?



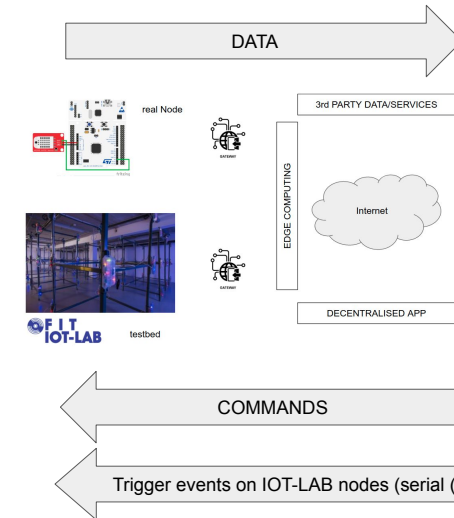
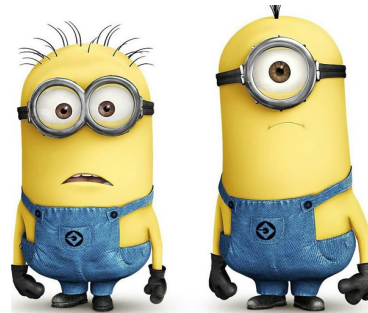
Your final deployment, in its essence, should look similar to the one depicted in the picture.

- The “real node” is used to prove your ability of integrating the specific sensors envisioned in your application scenario
- The lot-lab deployment is used to prove your ability to scale-up your solution into an ecosystem of nodes organised in a network and to evaluate their performance

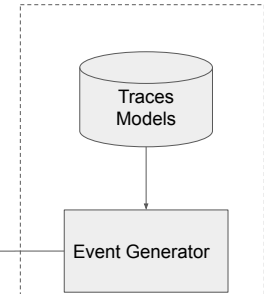
Problem

Since we cannot integrate arbitrary sensors (e.g. stress sensor) in the lot-Lab deployment, and in any case we do not have the ability of generating specific events to trigger interesting monitoring activities (e.s. apply a force to a sensor)

how can we prove the efficiency/effectiveness of the proposed solution in “realistic” application scenarios?



A possible approach



A firmware skeleton

BOARD=iotlab-m3 make all

```
#include <stdio.h>
#include <string.h>
#include "shell.h"

int trigger(int argc, char **argv)
{
    printf("triggered %s\n", argv[1]);
    (void)argv;
    return 0;
}

static const shell_command_t commands[] = {
    { "trigger", "trigger an event", trigger },
    { NULL, NULL, NULL }
};

int main(void)
{
    puts("Trigger events");
    char line_buf[SHELL_DEFAULT_BUFSIZE];
    shell_run(commands, line_buf, SHELL_DEFAULT_BUFSIZE);
    return 0;
}
```

/home/andrea/Documents/University/teaching/IoT/IoT-Lab/RIOT/examples/triggers

Experiment trigger #256238

User vitalett

Submitted 2021-03-29 22:23:29

Started 2021-03-29 22:23:30

Duration 0 minutes (0%) of 20 minutes






Nodes 1

State **Running**





Stop






Download

Actions on selected nodes

Nodes	UID	Firmware	Monitoring	Deployment	Actions
m3-101.grenoble.iot-lab.info	9181	Trigger.elf		Success	    
<pre>main(): This is RIOT! (Version: 2021.01) trigger events > trigger temperature trigger temperature triggered trigger:temperature ></pre>					

Remember to close the terminal

Nodes	UID	Firmware	Monitoring	Deployment	Actions
m3-101.grenoble.iot-lab.info	9181	Trigger.elf		Success	    

Nodes	UID	Firmware	Monitoring	Deployment	Actions
m3-101.grenoble.iot-lab.info	9181	Trigger.elf		Success	    

```
andrea@andrea-HP-EliteBook-830-G5:~$ ssh vitalett@grenoble.iot-lab.info
Linux grenoble 4.19.0-13-amd64 #1 SMP Debian 4.19.160-2 (2020-11-28) x86_64
Welcome FIT IoT-LAB users
```

```
vitalett@grenoble:~$ nc m3-101 20000
trigger humidity
trigger humidity
triggered trigger:humidity
>
```

```
andrea@andrea-HP-EliteBook-830-G5:~$ ssh -L 20000:m3-101:20000 vitalett@grenoble.iot-lab.info
Linux grenoble 4.19.0-13-amd64 #1 SMP Debian 4.19.160-2 (2020-11-28) x86_64
Welcome FIT IoT-LAB users
```

```
andrea@andrea-HP-EliteBook-830-G5:~$ nc localhost 20000
trigger ciao
trigger ciao
triggered trigger:ciao
> ^C
andrea@andrea-HP-EliteBook-830-G5:~$
```

```
#!/bin/bash
for i in 1 2 3 4 5
do
    echo "Trigger $i times"
done
```

<https://unix.stackexchange.com/questions/332163/netcat-send-text-to-echo-service-read-reply-then-exit>

iot-lab.info/legacy/tutorial/sa...legacy/aggregator/index.html

2) WhatsApp Imported Scimgao Jour... didattica Trello

NEWS PLATFORM DEV CENTER COMMUNITY GET STARTED ACTIVITY Access the Testbed

Nodes Serial Link Aggregation

Difficulty: Medium

Duration: 30 minutes

Prerequisites: Submit an experiment with M3 nodes using the web portal / Experiment CLI Client

Description: This document shows how to use the `serial_aggregator` tool on the `ssh frontend`. You will learn how to quickly get the output of all your experiment nodes and how to send messages to some of them.

Description

Each node serial link can be accessed via a top socket to the node url on port 20000. Here is an example using net cat on node M3 with id 10 on Grenoble site involved in your running experiment :

```
user@grenoble:~$ nc m3-10 20000
```

When running experiments with many nodes, you may want to access the serial link of all the nodes of your experiment at once. The presented tool `serial_aggregator` should relieve the pain of having multiple terminals each connected to one node.

You will have all the nodes output in one terminal, prefixed by a timestamp and the node identifier.

```
5 hauser@grenoble:~$ serial_aggregator
1417086256.4378023[Aggregator] started
1417086256.4377007[m3-11]   [iotlab_uid] == 9971
1417086256.439846[m3-11]   [platform_uid] == 9971
1417086256.439846[m3-11]   [node_from_uid] == 31008
1417086256.440103[m3-11]   [node_atc] == m3-11
1417086257.988910[m3-10]   [iotlab_uid] == 8770
1417086257.989130[m3-10]   [platform_uid] == 8770
1417086257.989130[m3-10]   [node_from_uid] == 01000
1417086257.989130[m3-10]   [unknown node]
1417086262.4390683[Stopping]
```

⚡ Stopping using "Ctrl+C"

iot-os.org/api/group_drivers_saul.html

Apps (42) WhatsApp Imported Scimgao Jour... didattica Trello

RIOT The friendly Operating System for the Internet of Things Main Page Related Pages Modules Data Structures

[S]ensor [A]ctuator [U]ber [L]ayer

Drivers

Generic sensor/actuator abstraction layer for RIOT.

Detailed Description

Generic sensor/actuator abstraction layer for RIOT.

SAUL is a generic actuator/sensor interface in RIOT. Its purpose is to enable unified interface. Each device driver implementing this interface has to expose a set of predefined functions. Each device has further to expose a name and its type. This information can be used for a The SAUL module enables further the automated initialization of preconfigured actuators/s SAUL drivers may rely on being called from a thread, and often block for short amounts of

Todo:

So far, the interface only supports simple read and set operations. It probably needs to

See also

SAUL registry

/home/andrea/Documents
/University/teaching/iot/Io
t-lab/RIOT/examples/saul

<https://github.com/RIOT-T-OS/RIOT/tree/master/examples/saul>

Experiment test #256239

User **vitalett**

Submitted **2021-03-29 22:42:24**

Started **2021-03-29 22:42:26**

Duration **1 minute (5%)** of **20 minutes**

Nodes **1**

State **Running**

Stop Download

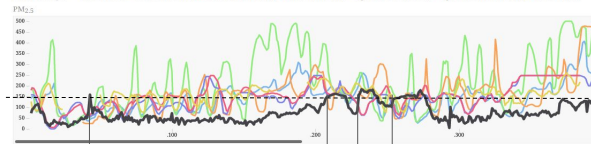
Actions on selected nodes

Nodes	UID	Firmware	Monitoring	Deployment	Actions
m3-96.grenoble.iot-lab.info	b468	saul_example.elf		Success	

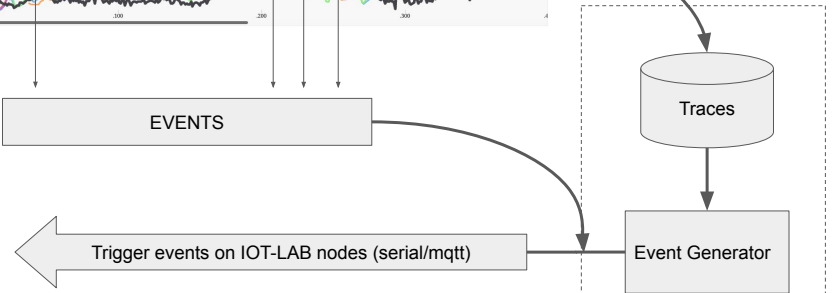
```
#6 SENSE_TEMP lps33lap
#7 SENSE_ACCEL lsm303dlhc
#8 SENSE_MAG lsm303dlhc
> saul
saul
ID Class Name
#0 ACT_SWITCH LED(red)
#1 ACT_SWITCH LED(green)
#2 ACT_SWITCH LED(orange)
#3 SENSE_LIGHT ls129020
#4 SENSE_GYRO l3g4200d
#5 SENSE_PRESS lps33lap
#6 SENSE_TEMP lps33lap
#7 SENSE_ACCEL lsm303dlhc
#8 SENSE_MAG lsm303dlhc
> saul read 6
saul read 6
Reading from #6 (lps33lap[SENSE_TEMP])
Data: 34.24 °C
```

An example <https://aqicn.org/forecast>

PM2.5 (in AQI) ■ Observation ■ Silam Asia PM2.5 ■ Silam World PM2.5 ■ Cams NRT PM2.5 ■ Sprinters Asia PM2.5 ■ Sprinters World PM2.5 ■ MPM China



Assume you have to monitor PM2.5 air pollution and you are interested in events generated when observations are above 150



DATASET REAL TRACES

There are 3 **mobility traces** datasets available on data.world. Find open data about mobility traces contributed by thousands of users and organizations across the

TOP OPEN-DATA SOURCES

mobility-traces-of-test-cube

A Community Resource for Archiving Wireless Data At DataMouth - Updated 8 years ago

Dataset of mobility traces of Test Cube in Rome, Italy

14 0 Comment

mobility-traces-9f-trails

A Community Resource for Archiving Wireless Data At DataMouth - Updated 8 years ago

Dataset of mobility traces of Test Cube in San Francisco, USA

29 0 Comment

pocket-mobility-trace-recorder

A Community Resource for Archiving Wireless Data At DataMouth - Updated 8 years ago

Dataset of mobility traces collected by Pocket Mobility Trace Recorder devices at University of Milano

0 0 Comment

RAWDAD

A Community Resource for Archiving Wireless Data At DataMouth

2020-05-26 emulogee-emartions

Captured ZigBee packets from commercial smart home devices.

Contributed by Dimitris-Georgios Alexandris, Muthumitha Harshankar, Michael Weber, Patrick Tagus.

10 0 Comment

2020-04-08 hms-hmsconcepts

Dataset of device context information (e.g., ambient audio) collected by multiple devices in different environments

Contributed by Michal Formichev, Max Maass, Lars Almon, Alejandro Molina, Matthias Hollick.

10 0 Comment

2020-02-18 hms-hmsconcepts

Dataset for evaluation of surveillance detection

Contributed by Michael Heus, Aaron Yi Ding, Jing Oit.

10 0 Comment

2019-10-30 lucymobile

Community RF Sensing via iPhones for Source Localization and Coverage Maps

Contributed by Emmanuel Aïmeur, Agnès Bellas.

10 0 Comment

2019-09-16 hms-hmsconcepts

GPS traces collected from a team of firefighters during a forest fire exercise

Contributed by Kris Agalar.

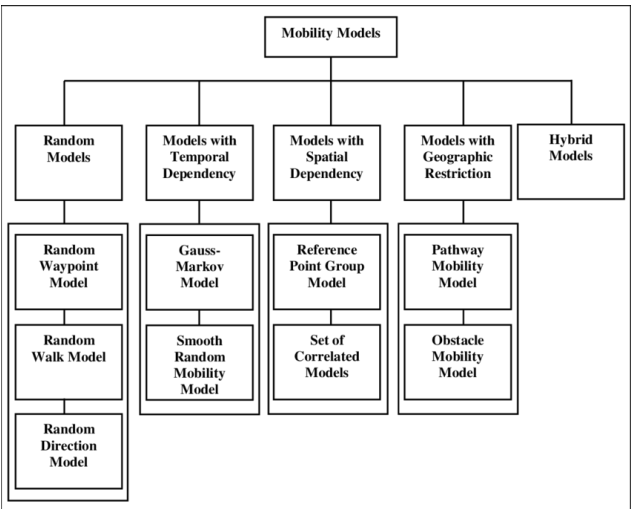
10 0 Comment

2019-08-25 hms-hmsconcepts

4G and 5G RAN monitoring data collected using the Eusilution 5G monitoring framework over FlanRAN

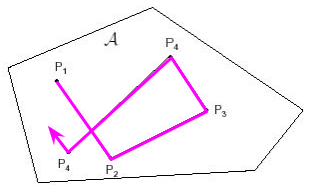
10 0 Comment

MODELS SYNTHETIC DATA



Random Way Point model:

- Each node moves along a zigzag line from one waypoint P_i to the next P_{i+1} .
- The waypoints are uniformly distributed over the given convex area, e.g. unit disk.
- At the start of each leg a random velocity is drawn from the velocity distribution.
- (in the basic case the velocity is constant 1)
- Optionally, the nodes may have so-called "thinking times" when they reach each waypoint before continuing on the next leg, where durations are independent and identically distributed random variables.



- Getting started
- Boards
- Deployment
- Grenoble
- Lille
- Lyon
- Saclay
- Strasbourg
- OS
- Tools

Lille

On the Lille site, the nodes are deployed in our two buildings at Inria Lille – Nord Europe. The most part is spread across the **A building**, usefull for large network and multi-hop experimentations. An additional part is deployed in a structure called **Le Cube**, in the B building, with a variety of boards.

server [lin.us lab info](#)
boards [BBC microbit](#) [IoT-LAB M3](#) [Mimicchip SAMR21](#) [ST-B-L0722-LRWAN1](#) [Zolertia Firefly](#)

Resources

- In the **A building**:
 - 256 IoT-LAB M3
 - 11 Zolertia Firefly
- A set of different boards in **Le Cube**:
 - 5 BBC microbit
 - 5 IoT-LAB M3
 - 5 Microchip SAMR21
 - 5 ST-B-L0722-LRWAN1
 - 5 Zolertia Firefly

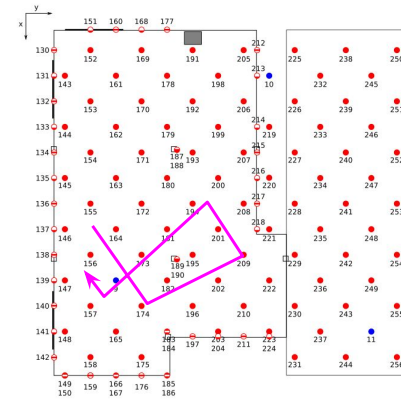
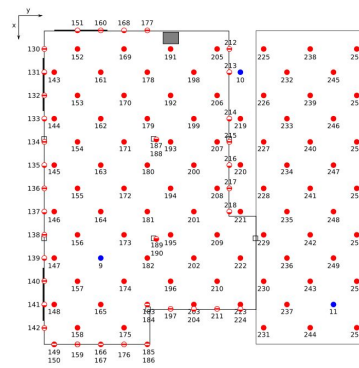


Topology in the A Building

Lille testbed is deployed at a building scale, over the three floors of our Inria building, through offices, corridors, meeting or storage rooms, in addition to a dedicated room. The deployment zones over the building are marked in blue in the figure below. See the video below to visit the deployment across the building.



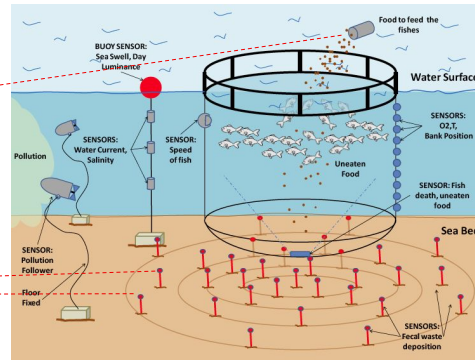
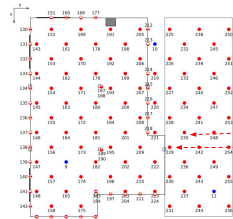
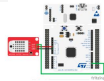
- Ground floor: m3-[1-45], firefly-[1-3]
- First floor: m3-[46-111], firefly-[4-6]
- Second floor: m3-[112-129], firefly-[7-8]
- Dedicated room (2nd floor): m3-[130-256], firefly[9-11]



Remember ... make reasonable assumptions and always support your claims with quantitative evidences!



REMEMBER



ASSUMPTION: Simulation of underwater links with wireless ... I'm perfectly aware this is a simplistic assumption because ... but Then I consider it as an upper bound on performance!