ESCUELA POLITÉCNICA SUPERIOR DE MONDRAGON UNIBERTSITATEA
MONDRAGON UNIBERTSITATEKO GOI ESKOLA POLITEKNIKOA
MONDRAGON UNIVERSITY FACULTY OF ENGINEERING

Partial release of the project

GRADO EN INGENIERÍA EN INFORMÁTICA
INFORMATIKAKO INGENIARITZA GRADUA
DEGREE IN COMPUTER ENGINEERING

Título del Trabajo *Lanaren izenburua* Project Topic

RUNTIME VERIFICATION FOR SPATIO-TEMPORAL PROPERTIES OVER IOT NETWORKS

Autor *Egilea* Author	OIHANA GARCIA ANACABE
Curso *Ikasturtea* Year	2021/2022

Nombre y apellidos del autor
Egilearen izen-abizenak
Author's name and surnames
GARCIA ANACABE, OIHANA

Nombre y apellidos del/los director/es del trabajo
Zuzendariaren/zuzendarien izen-abizenak
Project director's name and surnames
EZIO BARTOCCI
ILLARRAMENDI, MIREN

Lugar donde se realiza el trabajo
Lana egin deneko lekua
Company where the project is being developed
TU WIEN

Curso académico
Ikasturtea
Academic year
2021/2022

Table of contents

1. Description

IoT (Internet of Things) is the area of computer science that collects the challenges of connecting millions of smart devices and sensors and making them accessible via internet. This field is growing rapidly, it is estimated that by the end of 2022 there will be 42.56 billion connected devices [1].

Among the systems that can exploit an IoT infrastructure, a noteworthy category is Cyber Physical Systems (CPS), where physical systems are monitored and/or controlled by a computational core [2]. The following definition is the most famous one for the term “Cyber Physical Systems”:

"Cyber-Physical Systems are engineering, physical and biological systems whose operations are integrated, monitored, and/or controlled by a computational core. Components are networked at every scale. Computing is deeply embedded into every physical component, possibly even into materials. The computational core is an embedded system, usually demands real-time response, and is most often distributed. The behavior of a cyber-physical system is a fully-integrated hybridisation of computational (logical) and physical action."

(Helen Gill, US National Science Foundation)

Monitoring is an activity related to the wider category of Runtime Verification (RV), which purpose is to observe information from a system while it is operating and analyse the behaviour to detect if it satisfies or violates some properties [3].

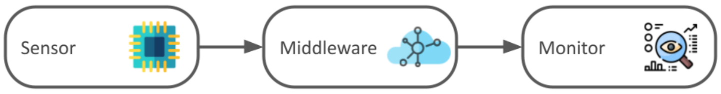


Figure 1-A Project outline

This project focuses precisely on the challenges when doing monitoring on CPS over IoT, and provides an implementation of a service to monitor data collected by sensors at runtime. It is closely related to some aspects of Helen Gill’s definition. The IoT devices are in the physical part where they are spatially distributed and networked. The data will be collected both across space and time. One main task of the project is to connect the sensors with the monitor so they can share information (i.e., networking). Finally, this data will be sent to MoonLight to monitor everything in real-time.

For this project, IoT sensors (Thingy52) and a monitor (MoonLight) are already provided. The resources will be studied and manipulated and, for the communication of these components, a middleware will be implemented. This monitor will be capable of monitoring at runtime. For the monitoring of spatio-temporal properties, logic-based specification languages such as STREL will be used. STREL permits to specify the requirements and to monitor them over a spatio-temporal trace.

2. Competences

Competences to be acquired in the Bachelor's Degree Final Project:

Table 1 Competences

Code	Description
G3I406	*Ariketa original bat indibidualki egitea eta unibertsitateko epaimahai baten aurrean aurkeztu eta defendatzea; ariketa Ingeniaritza Informatikoaren arloko proiektu bat izango da, izaera profesionalekoa, eta bertan ikasketetan barneratutako konpetentziak sintetizatu eta integratuko dira.*
	Individually carry out an original exercise and present and defend it in front of a university examining board; the exercise will consist of a computer engineering project of a professional nature in which the competencies included in the studies will be synthesized and integrated.
T1IT04	*Gaitasuna sistema, zerbitzu edo aplikazio informatikoak garatu eta exekutatzeko hardware eta software plataformak definitzeko, ebaluatzeko eta aukeratzeko.*
	Ability to define, evaluate and select hardware and software platforms for the development and implementation of computer systems, services or applications.
T1IT06	*Gaitasuna hardwarea, softwarea eta sareak integratzen dituzten sistema edo arkitektura informatiko zentralizatu edo banatuak sortzeko eta garatzeko.*
	Ability to create and develop centralized or distributed computer systems or architectures integrating hardware, software and networks.

3. Objectives

The objective of this project is to implement a demonstration methodology to monitor spatio-temporal properties using logic-based specification languages. To achieve this objective there are some crucial steps to do first:

- ☑ Analyze and comprehend STREL and the MoonLight monitor
- ☑ Identify the use case and data to monitor
- ☑ Set up an MQTT broker for the communication
- ☑ Establish an interface for the monitor and continuous streams
- ☑ Set up a converter from MQTT-data to moonlight-input streams
- ☑ Feed live data into moonlight and monitor

4. Product specifications and requirements

In this chapter the specifications and requirements of the project will be explained:

4.1. Resources and materials

The resources used for this project are the following ones:

Nordic Thingy52

The Nordic Thingy52 is an easy-to-use prototyping platform. This device has different hardware specifications, built around nRF52832 Bluetooth 5 SoC:

- Arm Cortex M4 32-bit, 64 MHz
- Configurable RGB LED and button
- Environmental sensors: Temperature, humidity, air pressure, air quality (CO2 and TVOC), color and light intensity
- Nine-axis motion sensing: Tap detection, orientation, step counter, quaternions, Euler angles, rotation matrix, gravity vector, compass heading, raw accelerometer, gyroscope, and compass data
- Sound: speaker for playing prestored samples, tones, or sound streamed over BLE (8 bit 8 kHz LoFi) and microphone streaming (ADPCM compressed 16 bit 16 kHz)
- Secure over-the-air device firmware upgrade (OTA DFU)
- Battery, rechargable, 1440 mAh
- Low power consumption
- 2.4 GHz transceiver, Bluetooth Low Energy
- Near field communication (NFC) support

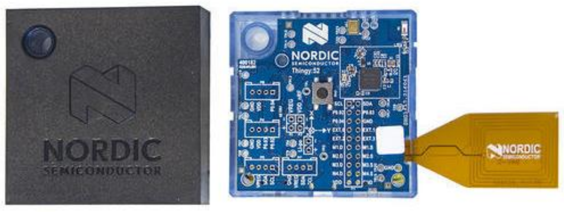


Figure 4-A Nordic Thingy52

Segger JLink

SEGGER J-Links are the most widely used line of debug probes on the market. Unparalleled performance, an extensive feature set, many supported CPUs and compatibility with popular environments.

- Programmer
- Debugger

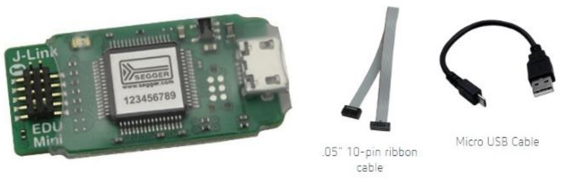


Figure 4-B J-Link EDU Mini and its cables

ESP 01

Espressif's ESP8266 ESP-01 delivers a highly integrated Wi-Fi SoC solution with efficient power usage, compact design and reliable performance in the Internet of Things industry. The features are the following ones:

- Integrated low power 32-bit MCU
- Integrated TCP/IP protocol stack
- 802.11 b/g/n WiFi 2.4 GHz, support WPA/WPA2
- Support STA/AP/STA+AP operation modes
- 10-bit ADC, SDIO 2.0, (H) SPI, UART, I2C, I2S, IR
- Remote Control, PWM, GPIO
- Deep sleep power < 10uA, Power down leakage current < 5 uA
- Wake up and transmit packet in < 2 ms
- Standby power consumption of < 1.0 mW (DTIM3)
- +20 dBM output power in 802.11b
- Operating temperature -40C – 125C

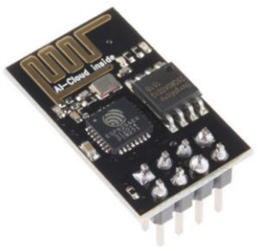


Figure 4-C ESP8266EX ESP-01 WiFi board

Zephyr

The Zephyr Project is a scalable open-source real-time operating system (RTOS) supporting multiple hardware architectures (more than 350 boards), optimized for resource-constrained devices, and built with security in mind.

The Zephyr projects are CMake-based. CMake is an open-source, cross-platform family of tools designed to build, test and package software.

Figure 4-D Zephyr and CMake logos

MoonLight

"MoonLight is a light-weight Java-tool for monitoring temporal, spatial and spatio-temporal properties of distributed complex systems, as Cyber-Physical Systems and Collective Adaptive Systems"[1]

- Supports the specification of properties written with the Reach and Escape Logic (STREL)
- Implemented in Java
  - Features MATLAB interface
  - Python Interface under development

IntelliJ IDEA

IntelliJ IDEA is an Integrated Development Environment (IDE) for JVM languages. It provides clever code completion, static code analysis and refactorings, doing the routine and repetitive tasks automatically.

Programming Languages

- Java11 is the language used in the monitor and middleware
- C/C++ is the language used for the Thingy52

[1] MoonLight: https://github.com/MoonLightSuite/MoonLight

Communication protocol MQTT

MQTT (Message Queuing Telemetry Transport) is a lightweight publish/subscribe protocol that transports messages between devices. This protocol is designed for IoT communication and low-bandwidth environments, which makes it suitable for this project.

Bluetooth Low Energy

Bluetooth Low Energy (Bluetooth LE) is a wireless personal area network technology. Compared to Classic Bluetooth, Bluetooth LE is intended to provide considerably reduced power consumption and cost while maintaining a similar communication range.

Mosquitto broker

Mosquitto is a lightweight open-source message broker. This is a suitable broker for this project and its IoT sensors.

4.2. Tests and trials

Gradle

Gradle is a build automation tool that will be used to compile, build, run tests and more.

SonarQube

SonarQube will be used to analyze, detect bugs, vulnerabilities, etc.

Trials

The monitoring method will be analyzed, studied and evaluated to establish a grade for the achieved results.

GitHub continuous integration

Continuous integration (CI) consists in committing code more often, in this way, the errors are detected sooner and the amount of code to debug when finding the source of an error is reduced. GitHub Actions offers workflows that can build the code in the repository and run the tests. These workflows can run on GitHub virtual machines.

4.3. Conditions for the implementation of the project

The development of the project may include the use of a server or additional sensors.

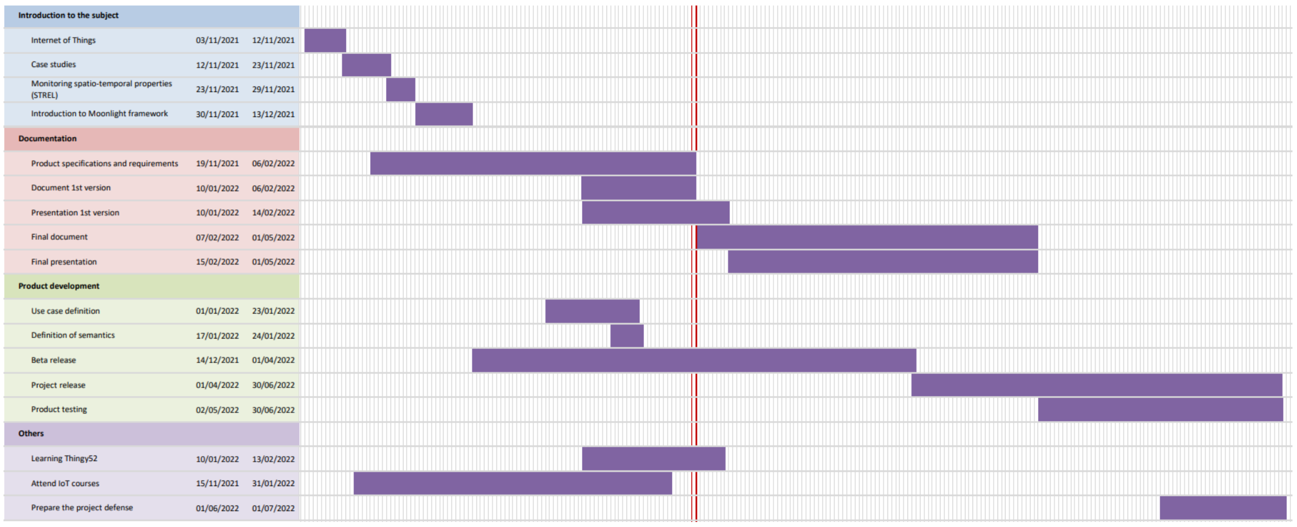
4.4. Legal aspects

General Data Protection Regulation (GDPR)

General Data Protection Regulation (GDPR) is a regulation on data protection and privacy in the European Union (EU). IoT devices tend to process personal data so data protection should be included in any IoT solution.

5. Project information

In this section, more specific information about the project is explained. The project duration is of eight months, from November 2021 to June 2022. To achieve the objective, the project has been divided into some tasks and scheduled to manage the work. The following Gantt chart describes the organization of the project.



Introduction to the subject		
Internet of Things	03/11/2021	12/11/2021
Case studies	12/11/2021	23/11/2021
Monitoring spatio-temporal properties (STREL)	23/11/2021	29/11/2021
Introduction to Moonlight framework	30/11/2021	13/12/2021
Documentation		
Product specifications and requirements	19/11/2021	06/02/2022
Document 1st version	10/01/2022	06/02/2022
Presentation 1st version	10/01/2022	14/02/2022
Final document	07/02/2022	01/05/2022
Final presentation	15/02/2022	01/05/2022
Product development		
Use case definition	01/01/2022	23/01/2022
Definition of semantics	17/01/2022	24/01/2022
Beta release	14/12/2021	01/04/2022
Project release	01/04/2022	30/06/2022
Product testing	02/05/2022	30/06/2022
Others		
Learning Thingy52	10/01/2022	13/02/2022
Attend IoT courses	15/11/2021	31/01/2022
Prepare the project defense	01/06/2022	01/07/2022

Figure 5-A Gantt chart: project schedule

5.1. Department

The project is held in Technische Universität Wien (TU Wien), Austria's largest research and educational institution in the field of technology and natural sciences. The group is the Cyber-Physical System (CPS) department.

5.2. Tasks

The following sections will explain the work done until now and the designed path to follow. The development of the work is being uploaded to two different public repositories[2].

5.2.1. Use case

In this world, there are many activities or events where this project might suit. IoT devices are growing up and so is the importance of monitoring/controlling. Cyber-physical systems can benefit several industries and organizations (e.g., smart cities, automotive, agriculture, health care). The use case chosen for this project is the next one:

Domotics / Smart Home Automation

A smart home is an accommodation provided with heating, lighting and electronic devices that can be monitored and controlled remotely. Through the IoT, it is possible to control and monitor the temperature, humidity, and all home devices from anywhere in the world. In this way, with proper monitoring and controlling, the user can minimize the wastage of electrical power and increase comfort and contentment [4].

In this use case, the project will focus on the CPS department.

[2] GitHub repositories: https://github.com/oihanagarciaa/GBL-IoTMonitoringSTREL & https://github.com/ennioVisco/MoonLight

- The Spatial Model will be constructed based on the map which can be found on the website of TU Wien:

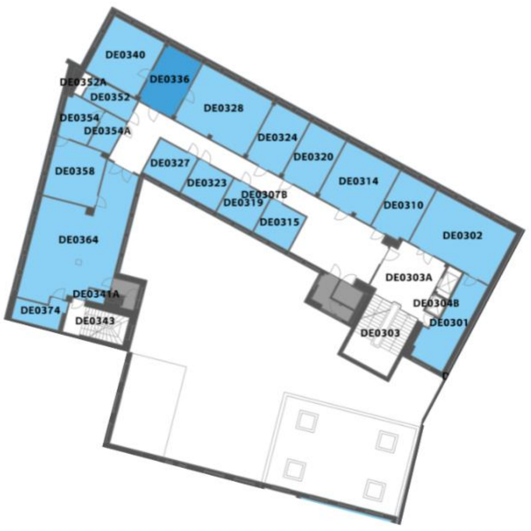


Figure 5-B Map of the department [5]

- Environmental data will be collected in the department:
    1. Temperature
    2. Humidity
    3. Air pressure
    4. Air quality (CO2 and TVOC)
    5. Light intensity
    6. Sound

- The monitoring is specification-based. To specify these spatio-temporal behaviors in a formal and human-understandable specification language, the logic used is STREL. This is the first approach of the specification properties of this project:

    - The environmental conditions must be adequate [6]:

        1. The temperature must be lower than 22ºC and higher than 15ºC
        2. Colour must be neutral/warm white
        3. CO2 less than X
        4. If the sound in one room is higher than X, the sound in the adjacent rooms should be less than X

5.2.2. Communication and monitor

In this section the project will be explained more deeply.

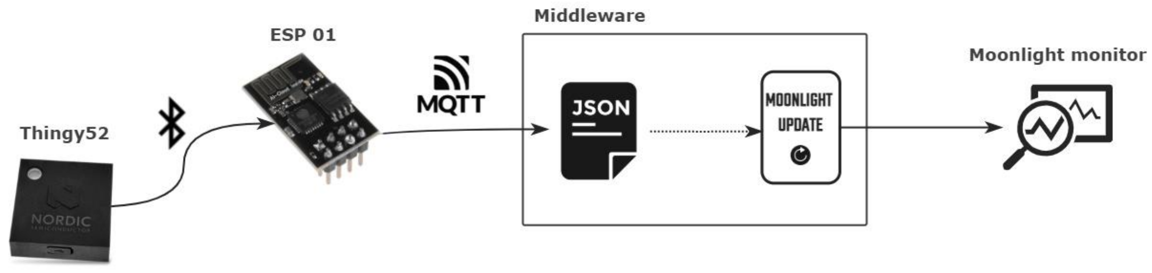


Figure 5-C Project communication overview

In the physical part, there are two types of devices. First, the Thingy52's sensors collect data from the environment. Then, the information is sent in a JSON file using Bluetooth Low Energy. The ESP01 receives the file, and it publishes to the MQTT broker, where the middleware is consuming the messages. The environment variables do not change frequently, for this reason, the sensor will not be uploading the data frequently either.

Regarding the monitoring segment, the middleware is responsible for converting the JSON files to a moonlight signal. The monitor will consist of an online monitoring (i.e., the monitoring is performed incrementally). When a JSON file arrives at the middleware, it is converted to an "update", the data class used in the online monitoring. After the preprocessing, Moonlight can monitor and check if the properties are satisfied. Moonlight must be able to get continuous streams, the implementation of an interface will be done to do that.

5.2.3. MQTT prototype

MQTT protocol will be in charge of transporting messages between the middleware and sensors. This protocol uses the publish/subscribe architecture, the sensor's role is publisher, and the middleware is the subscriber. The broker used in this project will be Mosquitto. This is the UML sequence diagram:

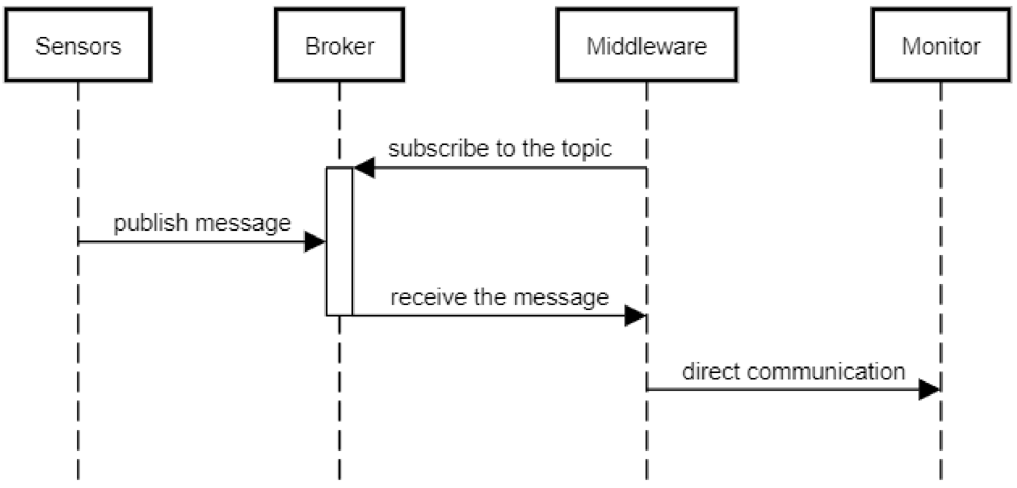


Figure 5-D MQTT UML Sequence Diagram

6. Problems and solutions

Throughout the project, some things have taken longer than expected and vice versa. In the work process I found some hitches that were no big issue and the next problem:

Table 2 Project problem and solution

Problem	I started setting up the Zephyr development environment on Windows, but I got an error that I could not fix. Zephyr SDK is not available on Windows, although there is a tutorial to set up everything on Windows, I kept hitting obstacles and the initialization process was dragging on.
Solution	I used my Linux Virtual Box Machine, where everything went more smoothly and I achieved to set up the environment and flash some projects to the Thingy52.

7. Conclusions and evaluation of the tasks carried out

In the first weeks, I read some articles to get familiarized with the topics involved in the project (i.e., Internet of Things, Runtime Verification, Cyber-physical systems and STREL).

While the use case was being decided, I started developing the moonlight interface and the middleware. For now, they receive some numbers from an online broker and monitor whether they satisfy some test specifications. Apart from setting up the base, these first steps have helped me to understand the classes, methods and workflow of the monitor. Now that I have the overview of the project defined, I have just started developing the Thingy52. Although I have just started to develop the hardware code, I have attended Prof. Ezio Bartocci's IoT courses, so it is not like starting from scratch, the course will help me a lot while doing this part.

In conclusion, I have the base and the structure of the project well defined. This should make the development go faster to achieve the objectives.

8. Future developments of the project

At the moment I will start with the Thingy52 and go climbing step by step until I reach all the objectives of the project. The project involves all the processes from the hardware to the monitor but adding other use cases will do the project more complete. The domotics/office use case satisfies the knowledge and the interaction that I will gain with the hardware, however, the properties to analyse are a bit poor. The temperature, humidity and air quality do not change much along space and time. For this reason, it will be interesting to add another use case, for example:

- **Wiener Linien use case**: Wiener Linien is the company that runs most of the public transit network in Vienna. They have an API that makes the public data accessible in real-time. This data is not only interesting to monitor, but also provides additional value to the project. It will be good to work with different data and to see that with minor changes the monitor can handle a completely different use case. Apart from that, there are more diverse specifications to use.

9. Contributions of the traineeships to studies

This project is helping to improve several aspects of my studies. On the one hand, I am working with different hardware, until now, I did not have the opportunity to use many devices. On the other hand, I am learning more about mathematical logic and expressions that I did not learn in the university.

Furthermore, I am understanding better the communication protocols and I am gaining experience while implementing this project. The programming language that I am mainly using is Java, a language that I already knew but am still learning. The other language I am using is C++ a language that I did not know much about.

10. Bibliography

[1] IHS, “Forbes,” 2016. [Online]. Available: https://www.forbes.com/sites/louiscolumbus/2016/11/27/roundup-of-internet-of-things-forecasts-and-market-estimates-2016/#6a558beb292d.

[2] L. Nenzi, E. Bartocci, L. Bortolussi, M. Loreti and E. Visconti, “Monitoring Spatio-Temporal Properties (Invited Tutorial),” in *Runtime Verification*, Springer International Publishing, 2020.

[3] C. Tsigkanos, M. M. Bersani, P. A. Frangoudis and S. Dustdar, “Edge-based Runtime Verification for the Internet of Things,” *IEEE Transactions on Services Computing,* 2021.

[4] K. Diponkar, K. Md, D. Tushar, M. Abdullah and M. Ahmmad, “Smart Home Automation System Using on IoT,” *International Journal of Scientific & Engineering Research,* vol. 11, pp. 697-701, 06 2020.

[5] TUWIen and indrz, “TU Wien maps,” [Online]. Available: https://tuw-maps.tuwien.ac.at/?q=DE0364#map.

[6] E. Sander, A. Caza and P. Jordan, “The physical work environment and its relationship to stress,” pp. 268-284, 06 2019.

[7] H. Gill, US National Science Foundation, 2006.