TOWARDS COMPUTATIONAL NOTEBOOKS FOR IOT DEVELOPMENT

Fulvio Corno, Luigi De Russis, and Juan Pablo Sáenz*

fulvio.corno@polito.it, luigi.derussis@polito.it, juan.saenz@polito.it

POLITECNICO **DI TORINO**



https://elite.polito.it/

CONTEXT AND GOAL

Internet of Things (IoT) systems are **complex** to develop due to the co-existence of various kinds of devices, protocols, architectures, and programming languages.

IoT programmers could benefit from **interactive doc-**

IOT NOTEBOOKS FEATURES

Upon the requirements stemming from the use case, we identified the **features** that, besides the ones that current computational notebooks support, an IoT **notebook** should provide:

Support multiple programming languages

umentation to edit and share textual explanations, code, dependencies, and terminal commands. **Computational notebooks** support the construction and sharing of computational narratives: they consolidate text, executable code, and visualizations in a single document.

Are **computational notebooks** a feasible approach to support the development of **IoT systems**, mainly with prototyping and experimenting purposes?

USE CASE

To address the research question, we picked up as use case a prototype IoT system named **Emergency Quest** (EQ) with a typical IoT architecture: devices, gateways, cloud services, and applications.

Some code snippets should remain on execution (always-on nature of IoT systems) The code must be executable on **external devices**

Support the specification and installation of **depen**dencies

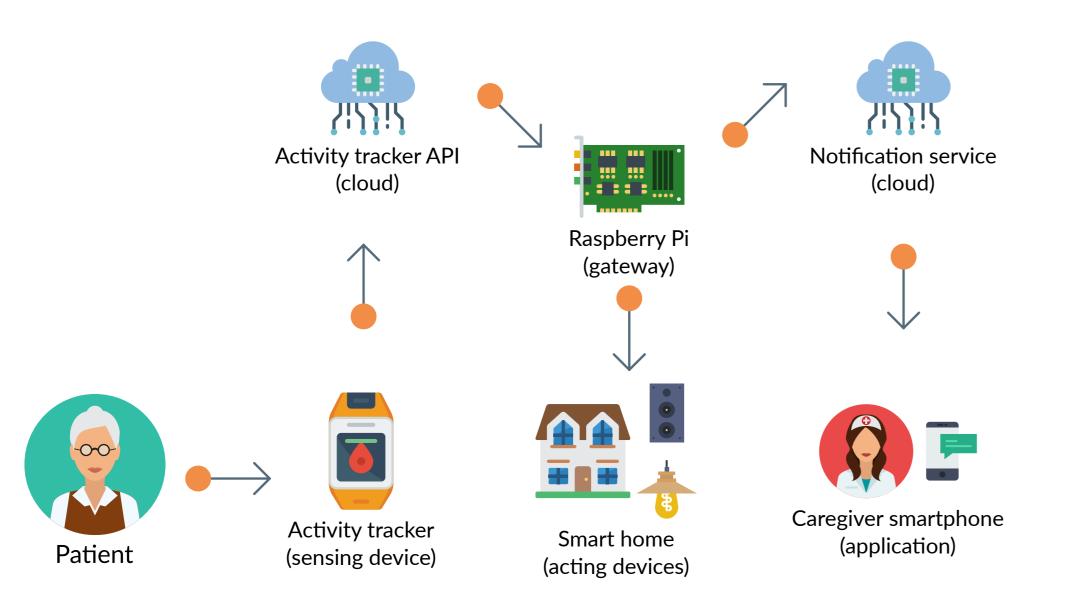
Support the integration of data coming from sensing devices or external platforms

VALIDATION

We implemented a prototypical IoT notebook by exploiting Jupyter and BeakerX.

B + S A I I A + H Run ■ C → Markdown ■ < Emergency Quest		H + ★ A	g
File Edit View Insert Cell Kernel Widgets Help	Trusted Python 3 O	File Edit View Insert Cell Kernel Navigate Widgets Help Trusted	Python 3 O
Jupyter Emergency Quest IoT project (autosaved)	Logout	CJupyter Integration between the gateway and smart home I (autosaved)	Logout
< > 🛄 192.168.0.12 C		< > 🛄 192.168.0.12 C 📋 -	- 🗇
:19 p. m. Dom 6 de ene.	? 0 26 % .	6:34 p. m. Dom 6 de ene.	• • 22 %

EQ aimed at improving the quality of life for mild dementia or Alzheimer patients. It was developed as the final project of a university course.



Through a wearable device, the system monitors the stress levels, location, and activities performed by the patient. If the system detects that the patient becomes agitated, it will react deploying a relaxing setup

IoT system developed in the Ambient Intelligence course at Politecnico di Torino

This is an IoT system aimed at improving the quality of life for people suffering from mild dementia or Alzheimer, while granting them greater autonomy and allowing their caregivers to be aware of their situation. By means of a wearable device the system can constantly monitor the stress levels, location and activities performed by the patient

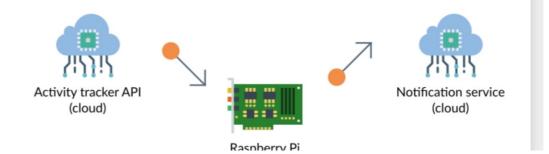
Description

Through a wearable device, the system monitors the stress levels, location, and activities performed by the patient. If the system detects, via the acceleration sensors of the bracelet, that the patient becomes agitated, it will react trying to calm him down through the deployment of a relaxing setup that includes music and lighting. Additionally, if the patient abandons a certain area, the system would react warning his caregivers by delivering notifications on their smartphones

Here below are presented the software and hardware components needed to implement and deploy Emergency Quest, as well as a diagram of the system architecture.

The complete implementation and deployment of each software component in the architecture are described into single notebook document.





FINDINGS

- Current computational notebooks partially satisfy the requirements of an IoT notebook.
- Special attention should be paid on how to smoothly execute the code snippets on low computational power external devices.
- New features should be introduced so users can handle **multi-file** projects and set the compulsory execution order of the snippets and their dependencies.

Once the gateway (Raspberry Pi 3 Model B+) determines that the patient has become agitated, it connects with the Philips Hue Smart home lighting and diffuses the lights. Here below is described step-by-step, first, the development of a Python controller, and later, the deployment of such controller in the Raspberry Pi.

Requirements

In order to complete this integration you must have:

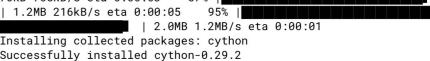
• Philips Hue hub: a Philips Hue bridge and know its ip address. Further documentation can be found at the official Philips Hue webpage (https://www2.meethue.com) • phue: a full featured Python library to control the Philips Hue lighting system

Installing phue

The development of this controller requires the installation of phue, a full featured Python library to control the Philips Hue lighting system.

In [3]: !python -m pip install --upgrade pip !python -m pip install cython !python -m pip install phue

> Collecting cython Downloading https://files.pythonhosted.org/packages/39/b1/2acbf92bl 3c817dc99a7588a6196629a0490b3f940b672136aa4d09f91ea/Cython-0.29.2-cp -cp36m-manylinux1 x86 64.whl (2.1MB) '6kB 705kB/s eta 0:00:03 57%



with music and lighting.

Additionally, if the patient leaves a certain area, the system would warn her caregivers through notifications on their **smartphones**.

The implementation of this system requires: the use of various languages; the deployment of components that keep running on the single-board **computer**; the installation of **dependencies**; and the interaction with third-party APIs.

Future work: Design and develop a complete IoTtailored notebook. Assess the benefits of a computational narrative in the context of IoT software development and prototyping through a **user study** with novice developers.