Internet of Things: Pallet Location Tracker

Mike Schumacher & Lucas Gehlen

***Abstract*— This document covers the IoT Project at Fontys Venlo of using a Pycom LoPy with GPS functionality as a Pallet Location Tracker, sending its location to a server installed on a Raspberry Pi and being accessible via a user interface. In this article we will specify our approach and the technologies that were used.**

I. Introduction

During our study at Fontys Venlo one of our subjects was Internet of Things(abbreviated as IoT). In this module, after learning a bit about working with IoT, we got the task to integrate our own IoT device into an application of our own. Initially, we chose our topic, which was pallet tracking using a LoPy with GPS functionality. We then decided on the following use cases we wanted to implement:

- Send data from the device to a database
- Get the latest location of a device and display it in a user interface
- Check if a specific device is located within a specified area.

What technologies we used for each of these will be further specified in the next chapter.

II. Overview of Technologies

In this chapter all the technologies that were used during this project are listed and explained. These technologies were either suggested by our lecturer or technologies we think would enhance the project.

A. Node.js

Node.js is a server sided platform based on *JavaScript* and provides an environment that is light on resources. For this project *Node.js* was used to realize the server side of the Pallet Location Tracker and integrate the database. This is the base for all the use cases, providing an environment to host the database the location data gets sent to as well as providing a host which the location data can be gotten from, which is necessary for use case two and three.

B. Python

To program the Pytrack the language *Python* was used. *Python* is an interpreted, high-level, general-purpose programming language. Also required for the first case, as here the location data is actually sent.

C. MySQL

MySQL is an open-source relational database system. In this project *MySQL* is used for storing the GPS data from the pytrack in a database. *Node.js* also accesses the database to retrieve the data and makes it available via *REST API*. The final part of the first use case, as you cannot store data in a database without a database.

D. Raspberry PI

The *Raspberry PI* is a small computer that is used as a server in this project. On the *Raspberry Pi* a *MySQL* server and the *Node.js* server have been installed.

E. REST

REST, or REpresentional State Transfer is an architectural style for communication of IT-Systems over the web. The sever side of this project uses *REST* to provide several endpoints. This way the QT application can use these endpoints to get data from the server. This is the main part of both the second and third use case, specifying how exactly you get the data into the user interface.

F. LoRa Network

The Pytrack uses the *LoRa Network* to communicate with the The Things Network (abbreviated as TTN), which manages devices and enables you to use an API which you can use to get the data onto the *Node.js* server. *LoRa* has a high range and can communicate over a distance up to 10 km.

III. Architecture

In Figure 1 the architecture of the Pallet Location Tracker project is displayed. The architecture can be explain in a few steps. First the Pallet Tracker uses the *LoRa* Network to send new position updates every hour or after it registered an activity like movement to the TTN. The *Raspberry Pi* then updates the MySQL database every hour with data from the TTN and makes the data available over an *REST* endpoint. Finally the QT application can use the *REST* endpoint to get data from the pallet tracker and display it to the user.

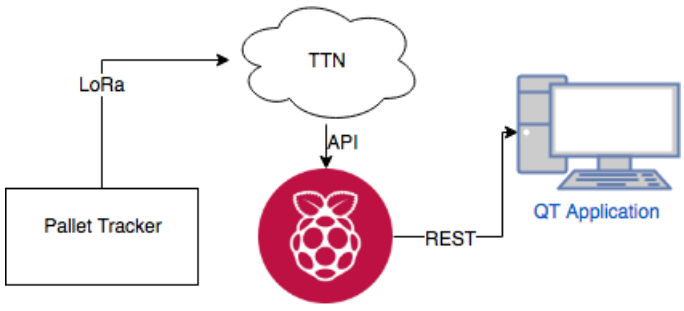

Fig. 1. Pallet Location Tracker Architecture

IV. State of the Project

V. Pytrack

The pytrack will be attached to a pallet and will send an update every hour with its current location. It will also update the location if its being moved automatically.
The following Pseudo Code shows how the pytrack works:

Algorithm 1 Pytrack Pseudo Code

```
1: procedure MAIN                          ▷ Start point of the pytrack
2:     while true do                                  ▷ infinity loop
3:         connect_to_lora();
4:         wait_for_gps_signal();
5:         send_data_to_api();
6:         sleep(1h);                      ▷ Go to sleep for 1 hour
```

The first thing the after the Pytrack has booted is to make a connection to the LoRa Network. After that it will activate the GPS sensor and wait for a GPS signal. As soon as the pytrack gets a GPS signal it will send the position over the LoRa Network to the TTN. If 40 seconds have passed and no GPS signal has been received, it will send the position "('None', 'None')". Whenever the Pytrack is not in use, it will go into sleep mode. It will wake up after 1 hour or if the activity sensor registers movement. If that is the case, it will try to get a GPS signal again and the cycle continues.

VI. Raspberry PI

The raspberry pi serves as a server for the Pallet Location Tracker. It will automatically save all the data from the TTN to a MySQL database, using an API that is integrated in the TTN. Furthermore it also provides a REST endpoint with 2 methods that is contacted by the QT application.
The first method will return the latest position of a specific tracker as well as the time when the tracker was at that location and the second method returns all the devices that are currently in use.

VII. QT Application

Using QT, a user interface has been created showing you every registered device and being able to get their latest locations.
The devices will be put into a dropdown menu with a button attached which gets the latest location of the selected pallet and displays it in a table.
Once a location has been gotten, the user is also able to check if the associated tracker is within a certain area. This area has to be specified using two coordinates which the user can input himself. Once entered, the user can click a button and a label will show if the selected pallet is within this area or not.
The user interface can be seen in figure 2.

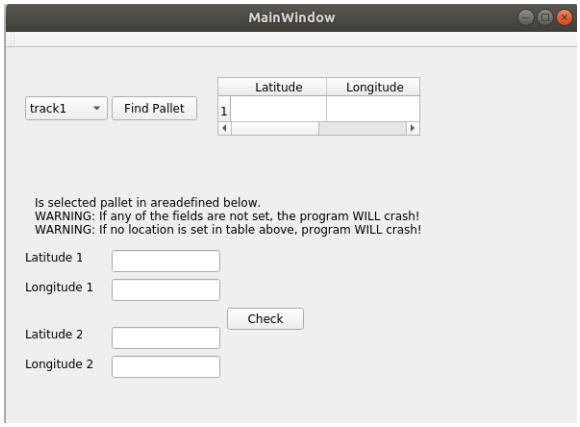

Fig. 2. User interface