# **Vehicle Crash Detection system**

# **Final Reoport**

## Name and Course

* Ashwath Shunti Mahabalagiri
* Connected Devices, CSYE 6530, Fall Semester, 2018

## Description

The aim of this project is to detect an accident/crash to the car using a sensor and notify the emergency authorities immediately to take the necessary action.

Whenever the sensor detects a crash to the vehicle when it is running, the device sends the data to the cloud through a gateway device. When the emergency unit responds to this call, a notification is sent directly to the driver and also to the police/ambulence. Gateway device will also send a notification to the owner of the vehicle when it recognises any crash notification.

Device Descriptions:

The sensor which is used for the testing is IR proximity sensor.

The constrained device used is Raspberry Pi

Gateway is any computer or laptop

The emergency authority is modeled as a cloud service provider, Ubidots education in this case.

## Code Repository

* URL or File Reference: <https://github.com/AshwathSM/CarCrashDetection/tree/master/VehicleCrashDetection>
* Demo video: <https://youtu.be/uTQbEtGFsXs>
* Listing of all modules :

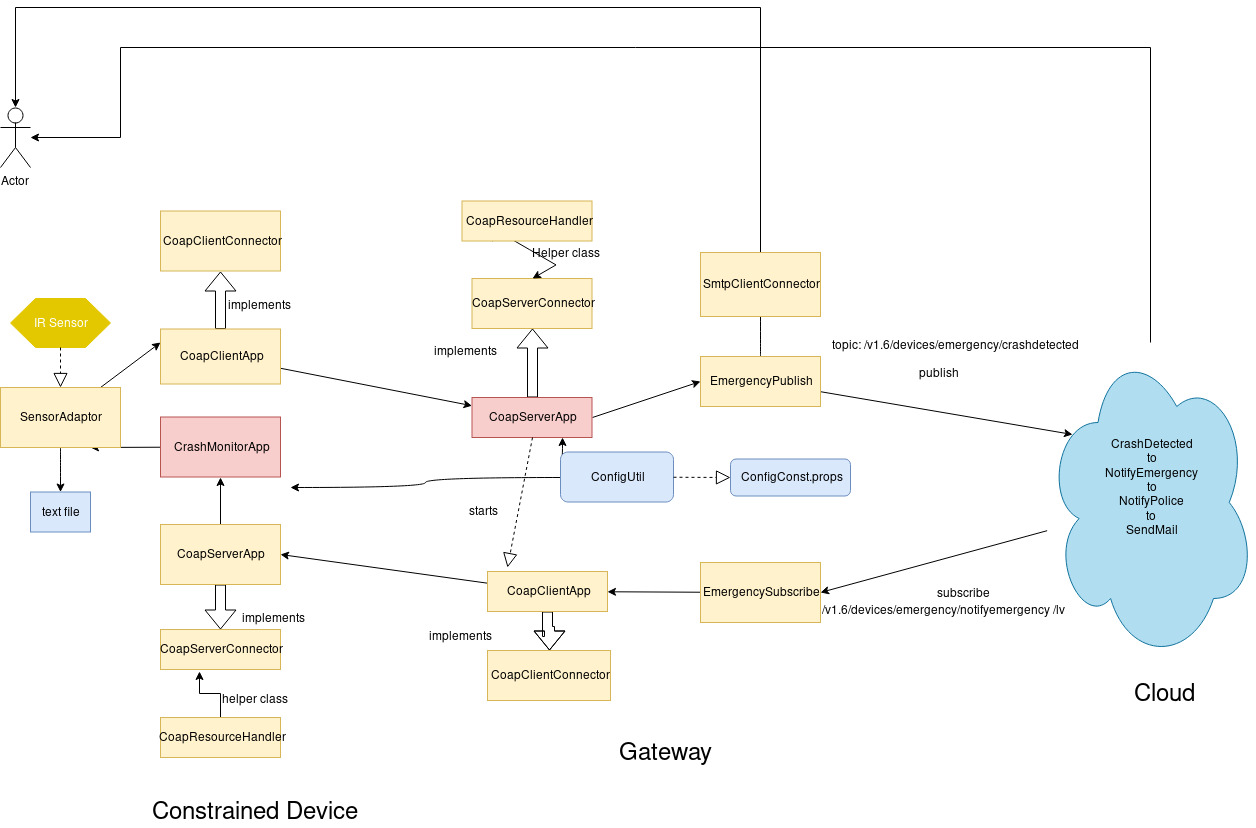
Raspberry pi modules:

* + CrashMonitorApp: Runs on Raspberry Pi. This will run in a loop and runs the coap client in case of a crash. This will also start a coap server in the raspberry pi so that any data from the gateway device can be processed.
  + CoapClientConnector: This module implements the “coapthon helper client” (refer https://github.com/Tanganelli/CoAPthon) to act as a client. This module will run whenever a crash detection is occurred and sends the data to the gateway device (CoAP server running in gateway)
  + CoapServerApp: This class instantiates the CoapClientConnector module and sends the crash information to the gateway. This is done by publishing to a specific topic.
  + CoapServerApp: This applcation instatiates the CoapServerConnector (which in turn implements “coapthon helper client”). This will wait for any data from the cloud and the data is displayed on a device/console/message so that the owner of the vehicle can read.
  + SensorAdaptor: This class will run in a loop and waits for any crash detection. It comes out of the loop inc ase of a detection and starts the coap client to notify the gateway

Gateway device modules:

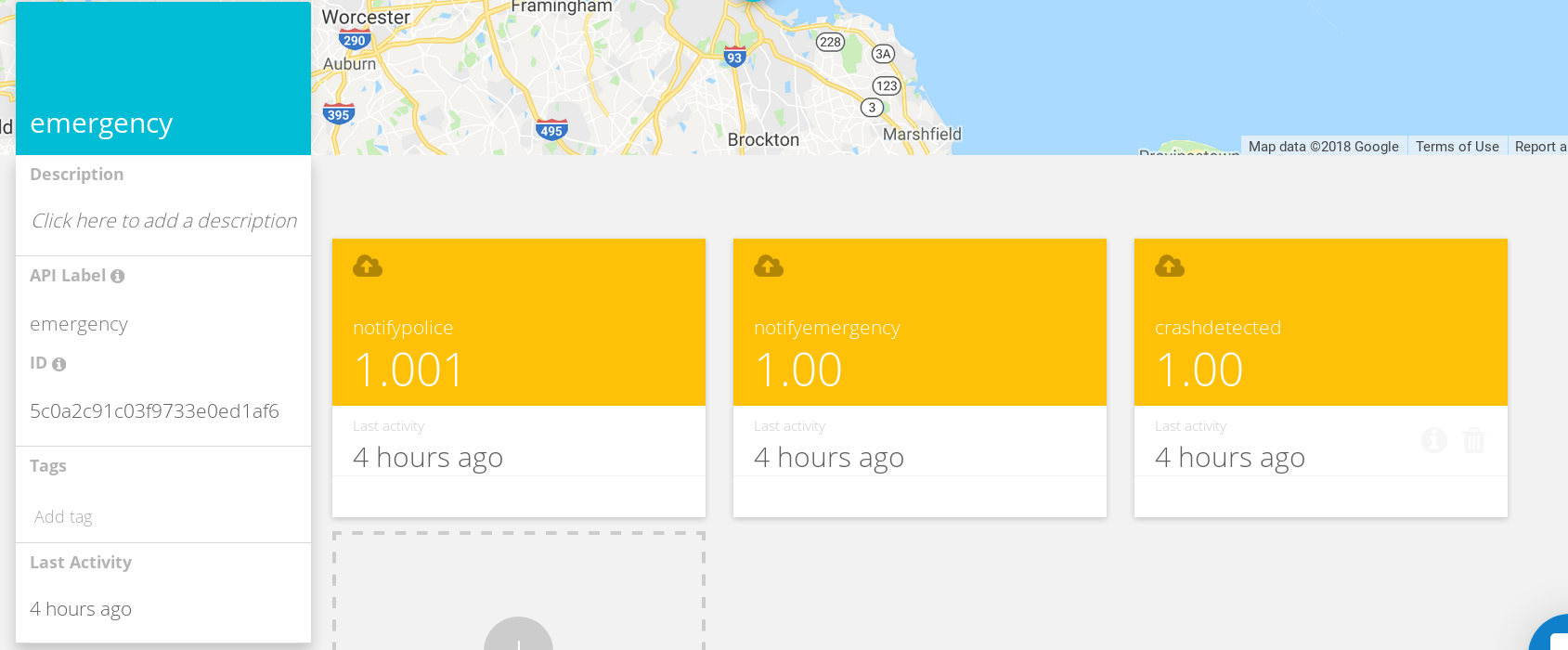
* CoapServerApp: This is the main applciation for the Gateway device. This will start a Coap server which will wait for any messages from the Coap Client running in the constrained device. This will also start EmergencyPublish (Mqtt Publish Client) to send the data to the cloud
* EmergencyPublish: This class will impliment paho Mqtt Client. And it connects to “things.ubidots.com” broker and publishes a message to the device on the Ubidots.(/v1.6/devices/emergency/crashdetected/).
* EmergencySubscribe: This class will implement paho Mqtt Client. It connects to “things.ubidots.com” broker and subscribes for a message on the topic /v1.6/devices/emergency/notifyemergency/lv. Whenever there is a message from the cloud it starts the coap client in the gateway
* CoapClientConnector: This class will implement the coapthon helper client to connect to the server running on the contrained device. This will transfer the data from the MqTT subscibe client to the Constrained device.
* CoapResourceHandler: This module (both in constrained device and gateway) helps in performing the request service actions for Coap Servers.

## Block Diagram



Snapshots:

3 Variables on Ubidots:



Events:

