# Nomenclature

433MHz wireless radios running at 433 MHz

RPI#1 Raspberry PI 3 acting as gateway collecting harvesters data and transform into SMOOL-SOFIA messages

RPI#2 Raspberry PI 3 acting as SMOOL internal server and as subscriber of SMOOL data. This device is also the link for exchanging data among other SCOTT elements (the CMW and the SFTM mainly)

GPIO input/output pins

CMW Indra’s gateway for communications in the shunting yard

MCU general purpose microcontroller dedicated to connect “things” to “networks”. This chip is useful in the harvester, to operate and to predict future energy intakes, and in the RPI#1, to isolate and filter wireless radio communication (insecure) into secure serial data to the GPIO pins

SMOOL middleware to create Internet of things ecosystems using semantic technologies. This software is based on SOFIA (UE funding) project specifications

SIB Semantic Information Broker. The SMOOL server.

KP knowledge processor. A SMOOL client sending or receiving data in OWL format

# Installation

## On train-wagons:

* hardware:
  + harvester unit
  + 433 MHz transmitter with monopole antenna.
* software: energy prediction algorithm, security module, etc.

## On shunting yard facilities:

* hardware:
  + raspberry PI 3
  + 433MHz receiver (one per area) with dipole antenna.
  + Microcontroller for receiving and cleaning wireless data and output to the serial port
* software:
  + scott-serial: read info from harvesters, check security and send harvester ID by IPC-socket.
  + SOTTProducer: get harvester IDs processed previously from scott-serial, and transform into SMOOL SSAP messages to be sent into the IoT ecosystem.

## On central shunting yard facility:

* hardware:
  + raspberry PI 3
* Software:
  + SMOOL server (SIB): IoT broker for SMOOL communications. Since shunting yard is a secured area, this server is better not to be in cloud (so communications are only in the shunting yard network). The SMOOL server can manage other messages non related to harvester positioning (generic alarms, new subscribers, etc)
  + SCOTTConsumer: client to receive harvester alive signals as semanic concepts by using subscription to the SMOOL IoT ecosystem.
  + scott-mqtt: Send harvester data taken from SCOTTConsumer, to the SFTM services, by using one of the CMW MQTT brokers.

Nota: meter aqui la parte no critica de los README.md de los diferentes proyectos

# Configuration

See README.md on each project for detailed configuration.

* Harvester: preconfigured with unique ID when burning code in the MCU
* scott-serial: GPIO serial pins, socket client port
* SCOTTProducer: socket server port, SMOOL server address:port
* SMOOL server (SIB): internal TCP address and port
* SCOTTConsumer: SMOOL server address:port, socket client port
* scott-mqtt: security connection params to CMW, socket server port

# Usage

Assuming the simplest case: 1 harvesting unit, 1 raspberry to control area and 1 raspberry to provide SMOOL capabilities and communication with external services.

1. Raspberry PI #1: start Serial and Producer to receive the ‘alive’ unit from the harvesting unit.
2. Raspberry Pi #2: start SMOOL Server, Consumer and Mqtt client.
3. The Harvester has enough energy to send alive signal (harvester ID).
4. The RPI #1 has a range limit, so location and precision can be deduced.
5. The GPIO Serial reads the harvester ID. This id has been extracted an cleaned from the MCU attached to the 433 radio.
6. Security check is performed based on a rule. For instance if SUM(id) != 10 id is rejected.

harvester ID is sent via local socket to SCOTTProducer. Now SMOOL technology takes control of the message flow.

1. SCOTTProducer assembly the ID into a SSAP message to communicate with other clients with SMOOL capabilities. The messages are sent containing OWL triples so other systems could consume data from harvesters even if they do not now what kind of device is sending data.
2. The message arrives to SMOOL server (SIB). This element acts as a semantic broker to extract what kind of information is arriving and what are the clients subscribed to this information or to parent hierachical information (example: some clients can subscribe to position information only while others can subscribe to any information a device can generate). A new message is sent to all clients subscribed directly or indirectly to that information.
3. The message from the SMOOL server arrives to the Consumer client. This message contains useful data and metadata to be digested. Once the harvester ID and position are extracted, the resulting data can be simplified and send via Socket to another client dedicated exclusively to communication with third parties. Here the SMOOL technology ends its flow for this message. Since the consumer is a SMOOL client, it can be also subscribed to other concepts (security concepts, alarms, etc) so it can react to other messages different than the position data.
4. The Scott-Mqtt application gets the position data. The ID of the harvester is matched to the wagon ID or to the container ID where the harvester was attached. A JSON message in the canonical model specified from Indra is sent to the

AÑADIR UN MENSAJE DE EJEMPLO DE CANONICAL

central server by using the nearest CMW visible to the client.

11. Finally the message is consumed by the SFTM services, listening to all position devices located in the shunting yard. Some of them are high precision, GPS location devices attached always to wagons while other elements are the autonomous, low power, disposable harvesters units attached to either wagons or containers or even dedicated cargo elements to be double-monitored (for instance because the load is an expensive item and needs to be tracked not only in the train trip but also in the origin and destination storage areas).

AÑADIR UN DIBUJO DEL FLUJO O VER SI LO TENEMOS EN ALGUN LADO de antes

<

# Connections

harvester-MCU-GPIO RPI#1: HACER DIBUJO CONEXION que tengo en el cuaderno

RPI#1-SMOOL: smool server is integrated in RPI#2 so RPI#1 & RPI#2 must be visible each other

SMOOL-RPI#2: Since SMOOL and KP receiving all data are in the same device, there is no network constraints

RPI#2-CMW: Ethernet connection to CMW in shunting yard

# Test

See README.md for detailed instructions on each application

The summarized procedure for testing each element is the following:

* Harvester: connect to RPI#1 and use `minicom` to check harvester IDs are arriving. Use command `sudo minicom -b 9600 -o -D /dev/serial0`. The energy efficiency algorithm installed in the harvester can be tested standalone in local PC since the core is C99 compliant.
* scott-serial: change serial address in `serial.js` (see README). Start `node index 1001` would emulate a harvester ID 1001 saying that this harvester is alive.
* SCOTTProducer: the IPC message is tested by opening a socket. Then, an SMOOL SSAP message is generated with `PresenceInformation` concept and sent to the IoT network.
* SMOOL server: test by connecting and sending any data from a generic client.
* SCOTTConsumer: if any `PresenceInformation` concept is being sending into SMOOL (not only from harvesters), this KP should react and check if it is a valid harvester or not. Then, when extracting data from the message and sending to the next element, the IPC socket to th the scott-mqtt is tested.
* Scott-mqtt: `node index 1001` emulates receiving data from Socket server and generate a JSON J131100 message (see INDRA’s canonical model and SFTM interfaces documents) that is sent to the CMW broker via MQTTs protocol.