Universitat Oberta de Catalunya

uoc.edu

Solució IoT de gestió de flotes pel transport frigorífic

David Moreno Cabruja
Màster universitari d'Enginyeria de Telecomunicació
Smart Cities

Juan Antonio Ortega Redondo
Carlos Monzo Sánchez

Gener de 2022

FITXA DEL TREBALL FINAL

Títol del treball:	*Solució IoT de gestió de flotes pel transport frigorífic*
Nom de l'autor:	*David Moreno Cabruja*
Nom del consultor/a:	*Juan Antonio Ortega Redondo*
Nom del PRA:	*Carlos Monzo Sánchez*
Data de lliurament (mm/aaaa):	*01/2022*
Titulació o programa:	*Màster universitat d'Enginyeria de Telecomunicació*
Àrea del Treball Final:	*Smart Cities*
Idioma del treball:	*Català*
Paraules clau	*IoT, AWS, MQTT*

Resum del Treball (màxim 250 paraules):

El control de la temperatura i la seguretat han estat dos punts clau en el transport i la distribució de les vacunes de la COVID-19 arreu del món. La disminució de preu dels components dels dispositius (sensors i actuadors), les xarxes sense fil més ràpides i l'augment de les capacitats d'anàlisi de dades estan fent que l'IoT esdevingui una peça clau en el sector de la logística i el transport.

En aquest context, s'ha implementat una solució IoT de gestió de flotes pel transport frigorífic. Mitjançant una comunicació bidireccional en temps real i l'ús de la plataforma Thingsboard, s'han pogut monitorar alguns paràmetres dels vehicles (temperatura, posició, velocitat...), gestionar alarmes i enviar ordres a aquests vehicles per controlar la temperatura del compartiment refrigerat. A més, s'ha utilitzat Amazon QuickSight per desenvolupar un quadre de comandament per a l'anàlisi forense de les dades IoT obtingudes.

Per una banda, s'ha desenvolupat un prototip amb tota la circuiteria i elements necessaris per capturar i enviar les dades dels vehicle mitjançant la tecnologia GPRS i el protocol de comunicació MQTT. Per altra banda, s'han simulat la resta de vehicles a partir de dades sintètiques per disposar d'una petita flota de vehicles i així comprovar el funcionament de la plataforma IoT creada.

Amb aquest projecte, s'ha obtingut una plataforma IoT completament integrada a AWS. Des d'una vessant pràctica, s'han analitzat les possibilitats que ofereixen les noves tecnologies i els sistemes de comunicació aplicats en l'IoT del sector del transport.

Abstract (in English, 250 words or less):

Temperature and security control have been two key points in the transportation and distribution of COVID-19 vaccines around de world. Decreased pricing of device components (sensors and actuators), faster wireless networks and increased data analytics capabilities are making the IoT a key part of the logistics and transport sector.

In this context, an IoT fleet management solution for refrigerated transport has been developed. Through two-way real time communication and Thingsboard platform, it has been possible to monitor some parameters of the vehicles (temperature, location, speed…), manage alarms and send commands to these vehicles to control the temperature of the refrigerated compartment. In addition, Amazon QuickSight has been used to develop a dashboard for forensic analysis of the obtained IoT data.

On one hand, a prototype has been developed with all the circuitry and elements needed to capture and send the vehicles data using GPRS technology and MQTT communication protocol. On the other hand, the rest of the vehicles have been simulated from synthetic data to have a small fleet of vehicles and check the operation of the developed IoT platform.

With this project, an IoT platform has been fully integrated into AWS. From a practical point of view, the possibilities offered by new technologies and communication systems applied in IoT of the transport sector have been analysed.

Índex

Llista de figures

Llista de taules

1. Introducció

En aquest capítol, es defineix la motivació que ha impulsat la realització d'aquest projecte, els objectius que es volen assolir amb la seva elaboració, el mètode seguit i la planificació del treball que s'ha establert per a la seva consecució.

1.1 Context i justificació del treball

Segons el document [1] publicat per l'Institut Internacional de Refrigeració (IIR), cada any en tot el món, uns 475 milions de tones d'aliments són malbaratats per la falta de refrigeració. De la mateixa manera, segons l'informe [2] realitzat per *IQVIA Institute*, la falta de control de temperatura i la logística necessària per mantenir la cadena de fred de forma ininterrompuda representa a la indústria farmacèutica un cost anual de 35 bilions de dòlars, aproximadament.

Precisament, un dels reptes més importants a nivell logístic que ha suposat la pandèmia de la COVID-19 ha estat el transport i la distribució segura de les vacunes als diferents centres de vacunació d'arreu del món. Han sigut necessàries una gran quantitat de mesures no només relacionades amb les línies aèries, companyies farmacèutiques, personal de duanes o policia sinó també quant als recursos que han estat necessaris: des de camions refrigerats i ultracongeladors fins a geolocalitzadors i alarmes.

Així doncs, el control de temperatura ha estat un dels punts clau del procés per tal que les vacunes arribessin al seu destí en perfecte estat. Encara que cadascuna té unes característiques diferents, la refrigeració ha estat una condició comuna i indispensable en totes elles per garantir la cadena de fred des de la producció fins al transport, l'emmagatzematge o el subministrament, i així conservar la qualitat, les propietats i les característiques de les dosis.

En el cas del transport terrestre, han estat necessaris grans camions frigorífics dotats amb equips de producció de fred. Sensors encarregats de registrar la temperatura i portes hermèticament tancades han permès garantir en tot moment que no es produïssin canvis de temperatura en el producte, mantenint així la cadena fred.

Però a banda de la refrigeració, la seguretat ha estat un altre aspecte clau en el transport de les vacunes de la COVID-19. La incorporació de geolocalitzadors a la mercaderia ha permès conèixer la posició i la temperatura de les vacunes en temps real des d'un centre de control. A més, mitjançant un sistema d'alarmes, també s'ha pogut alertar de possibles canvis de temperatura o intents de robatori.

Aquest és només un dels exemples més recents que demostra que la incorporació de la tecnologia de l'Internet of Things (IoT) en el sector del transport, permet garantir la seguretat i la integritat dels productes, conèixer en temps real el seu estat i la seva ubicació i, en general, optimitzar i simplificar la seva distribució.

Malgrat que es pot pensar que l'IoT ja està àmpliament establert en l'àmbit del transport i la logística, DHL en el seu informe *Next-Generation Wireless in Logistics* [3] posa de manifest que una nova generació de tecnologies de comunicació sense fils, més enllà del 5G, crearà noves oportunitats per a les indústries. La capacitat de supervisar, rastrejar i interactuar amb els actius mitjançant connexions sense fils farà que les cadenes de subministrament siguin més ràpides, flexibles, previsibles i autònomes, millorant així, l'eficiència operativa, la visibilitat i la qualitat del servei.

1.2 Objectius del treball

L'objectiu principal d'aquest projecte és desenvolupar una solució basada en IoT per a la gestió de flotes de vehicles de transport frigorífic per carretera. Mitjançant una comunicació bidireccional sense fils entre els nodes de la flota i una plataforma IoT es monitoraran en temps real paràmetres com la posició i la temperatura del remolc, es crearà un sistema d'avisos i alarmes i s'enviaran consignes de control als vehicles.

Per una banda, es pretén implementar un node físic amb tota la circuiteria i tots els elements necessaris per comprovar el funcionament en un entorn real. I, per altra banda, es pretén simular la resta de nodes per disposar d'una petita flota de vehicles i així poder crear una aplicació de gestió IoT complerta i funcional. Per tant, el sistema a desenvolupar constarà de les següents parts:

- Sistema de captació i actuació: aquest sistema, ubicat dins el vehicle, ha de permetre obtenir les dades d'interès del vehicle com la posició i la temperatura i enviar-les a la plataforma IoT. A la vegada, també ha de poder actuar sobre els diversos elements instal·lats als vehicles com l'equip de producció de fred.
- Simulador software: com que només s'implementa un node físic, la resta de vehicles de la flota seran simulats utilitzant un ordinador el qual serà l'encarregat de generar i enviar dades sintètiques com si es tractessin d'altres vehicles.
- Plataforma IoT: ha de permetre enregistrar les dades dels nodes (real i simulats), processar-les i presentar els resultats; monitorar les alarmes i les incidències i enviar comandes als vehicles com la consigna de temperatura del remolc. Per desenvolupar aquestes funcionalitats s'analitzaran les possibilitats que ofereixen les plataformes cloud IoT existents (Amazon Web Services, Microsoft Azure...)

A continuació, es detallen els objectius específics que s'abordaran al llarg d'aquest projecte dividits en les diferents parts de les quals consta la solució.

Sistema de captació i actuació:

- Muntatge d'un prototip per comprovar el funcionament en un entorn real.
- Obtenció de la posició del vehicle (latitud i longitud) i velocitat de forma periòdica.
- Obtenció de la temperatura interior i exterior del remolc de forma periòdica.
- Detecció d'obertura del compartiment frigorífic.
- (No prioritari) Control de l'equip de producció de fred segons consigna.
- Capacitat per descartar aquelles dades errònies o corruptes.
- Enviament de les dades a un servidor pel seu emmagatzematge i tractament.

Simulador software:

- Creació de diversos vehicles simulats de característiques similars al node físic.
- (No prioritari) Enviament d'altres dades com consum i nivell de combustible, accelerades i frenades i pressió dels pneumàtics.

Plataforma IoT:

- Presentació en una *dashboard* de les dades més rellevants de la flota.
- Visualització de la ubicació dels vehicles sobre un mapa en temps real.
- Visualització de les dades d'interès de cada vehicle de la flota en temps real.
- Visualització de l'històric de trajectes i dades dels vehicles.
- Gestió i monitoratge d'alarmes i incidències (temperatura fora dels límits establerts, velocitat màxima del vehicle superada...)
- (No prioritari) Enviament de consignes als vehicles com la temperatura.

1.3 Enfocament i mètode seguit

Aquest TFM parteix del projecte que es va realitzar anteriorment com a Treball Final en el Grau de Tecnologies de Telecomunicació anomenat “Sistema de seguiment de vehicles via web basat en GPS, GPRS i Arduino” [4]. El que es pretén ara és millorar i afegir noves funcionalitats al sistema assolint algunes de les línies de futur proposades, creant una solució IoT molt més completa i actualitzada a les necessitats actuals de la gestió de flotes aplicada, en aquest cas, al transport frigorífic.

És per això que, inicialment, es realitza un estudi de com ha evolucionat el control de la cadena de fred en el sector del transport, quin és l’estat actual de les tecnologies i els sistemes de comunicació utilitzats en logística relacionats amb IoT i quines són les principals problemàtiques que encara queden per resoldre. Per altra banda, també s’analitzen les plataformes IoT disponibles actualment al mercat, quin tipus d’informació aporten al client i els protocols de comunicació que utilitzen.

Tenint en compte això, en primer lloc es desenvolupa el node físic que va instal·lat a l’interior del vehicle. D’acord amb el projecte citat, els principals canvis i millores realitzats són els que es descriuen a continuació:

- Es realitza un estudi comparatiu de les principals tecnologies de comunicació per conèixer quina de les existents actualment és la més adequada per a un sistema d’aquestes característiques.
- La plataforma Arduino és reemplaçada per un microcontrolador més potent amb la possibilitat de gestionar tasques, establir prioritats o crear interrupcions. S’analitzen les diferents alternatives disponibles actualment al mercat que compleixin amb els requisits del sistema de captació i actuació que anirà instal·lat al vehicle.
- S’afegeixen sensors que permetin monitorar remotament alguns paràmetres dels vehicles com la temperatura o la detecció d’obertura del remolc.
- S’afegeix un equip de producció de fred que, encara que no és el que s’utilitza en els vehicles refrigerats, serveix per comprovar en un entorn real els canvis de temperatura i la capacitat de poder-la controlar dins uns límits establerts.

A continuació, com que només s’implementa un prototip, es simulen la resta de vehicles de la flota. Serà necessari generar i enviar dades sintètiques similars a les del node físic. Per això, s’implementarà una aplicació que permeti realitzar aquesta funció i compleixi amb les necessitats d’aquest treball.

El següent pas és determinar quin protocol de comunicació s’utilitzarà per enviar les dades des dels nodes. El canvi que es fa respecte el projecte anteriorment citat és el següent:

- El protocol de comunicació HTTP és substituït per un protocol més lleuger que pugui proporcionar una baixa latència i que sigui adequat per aplicacions que requereixen un baix consum d’energia i ample de banda. S’estudien les diferents possibilitats per determinar quina d’elles és la que més s’ajusta a aquestes necessitats.

Finalment, es valora quina de les plataformes IoT existents (Google Cloud, Amazon Web Services, Microsoft Azure...) és la que s'adapta millor als requisits d'aquest projecte per emmagatzemar, tractar i visualitzar gràficament les dades enviades pels nodes. Es verifiquen altres opcions existents al mercat i es valora la possibilitat de crear-ne una de nova que permeti implementar les millores que es volen realitzar partint del projecte anteriorment citat:

- Es crea un quadre de comandament (*dashboard*) principal i altres de secundaris per visualitzar simultàniament les dades d'interès de tots els nodes de la flota.
- S'afegeix una taula d'alarmes i incidències dels diferents vehicles de la flota.
- S'afegeix la capacitat de poder enviar comandes des de la interfície web als diferents dispositius per poder actuar remotament sobre els diversos elements instal·lats com la modificació de la temperatura de consigna de l'equip de producció de fred.

1.4 Planificació del treball

La planificació del treball s'ha estructurat en base als diferents lliuraments de les PAC, repartides al llarg de les 16 setmanes que dura el desenvolupament d'aquest TFM. Considerant que, de mitjana, es destinen unes 20 hores setmanals en la realització del treball, es requeriran unes 320 hores per a la seva correcta finalització.

Els primers 12 dies es destinen a realitzar la PAC1, una versió preliminar de la memòria que conté els capítols introductoris i l'índex provisional.

En els següents 14 dies es confecciona la PAC2, l'objectiu de la qual és redactar l'estat de l'art, corresponent al capítol 2 del projecte.

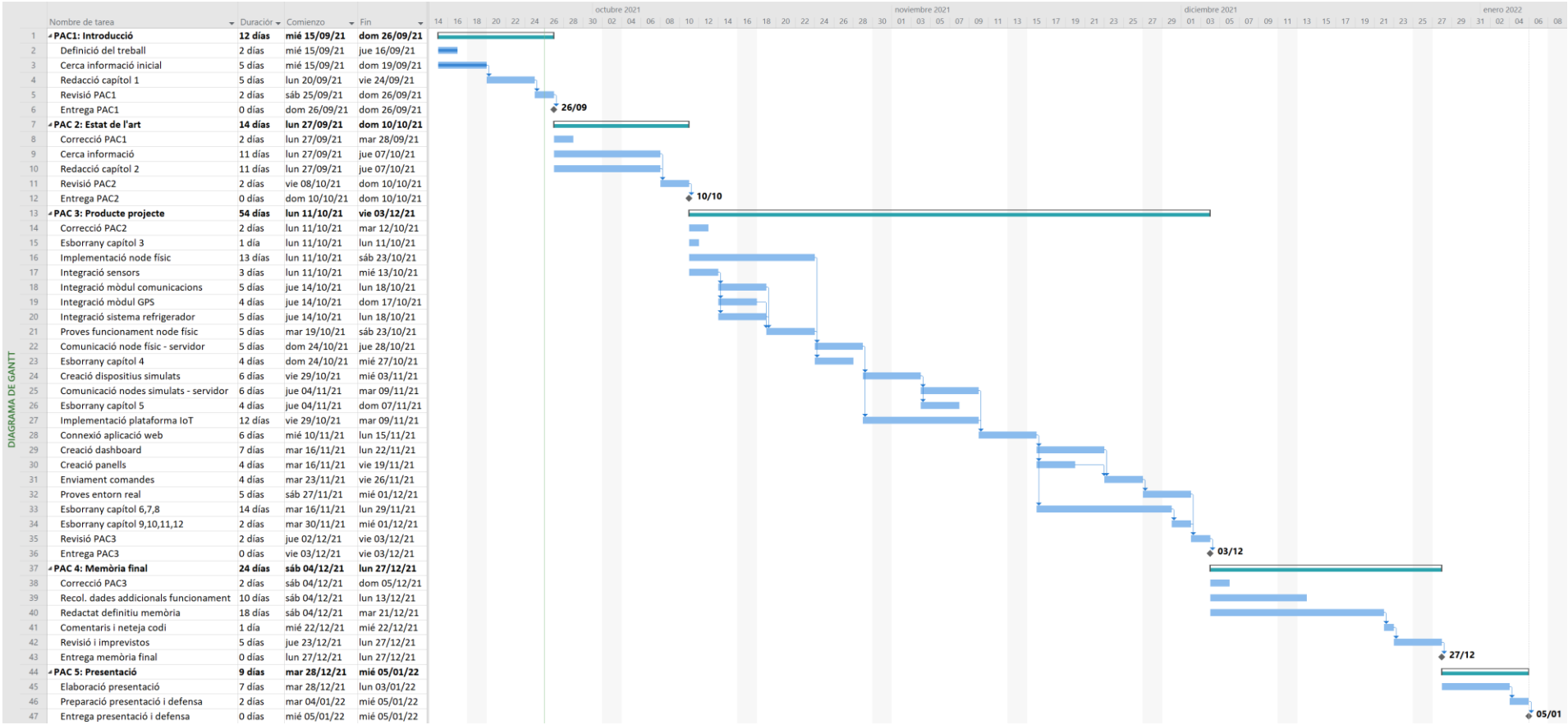
Per a la PAC3 es destinen 54 dies amb la finalitat d'obtenir el producte del treball. Aquesta fase és eminentment pràctica ja que es realitza el disseny i implementació del sistema i les primeres proves. No obstant, també es comencen a redactar de forma preliminar els diferents capítols de la memòria. Cal tenir present que en l'inici d'aquesta PAC ja cal disposar de tots els materials i components electrònics per poder començar amb el seu muntatge i programació.

Els següents 24 dies es destinen a realitzar la revisió i el redactat definitiu de la memòria i a comentar i netejar les línies del codi obtingut ja que en la PAC4 cal entregar la memòria finalitzada. Alguns dies d'aquesta fase també es destinen a recollir dades addicionals de funcionament del conjunt del sistema per completar l'apartat de resultats.

Finalment, els últims 9 dies s'utilitzen per realitzar la PAC5 en la qual cal confeccionar i preparar la presentació del TFM.

Cal destacar que dos dies abans de cada lliurament es destinen a revisar l'entrega i comprovar que es compleixen els criteris d'avaluació i dos dies després de cada lliurament es destinen a corregir el que procedeixi en funció del feedback rebut.

A continuació, es presenta el detall, en forma de diagrama de Gantt, de les diferents tasques que es duran a terme en cadascuna de les fases d'aquest TFM.



	Nombre de tarea	Duración	Comienzo	Fin
1	**PAC1: Introducció**	**12 días**	**mié 15/09/21**	**dom 26/09/21**
2	Definició del treball	2 días	mié 15/09/21	jue 16/09/21
3	Cerca informació inicial	5 días	mié 15/09/21	dom 19/09/21
4	Redacció capítol 1	5 días	lun 20/09/21	vie 24/09/21
5	Revisió PAC1	2 días	sáb 25/09/21	dom 26/09/21
6	Entrega PAC1	0 días	dom 26/09/21	dom 26/09/21
7	**PAC 2: Estat de l'art**	**14 días**	**lun 27/09/21**	**dom 10/10/21**
8	Correcció PAC1	2 días	lun 27/09/21	mar 28/09/21
9	Cerca informació	11 días	lun 27/09/21	jue 07/10/21
10	Redacció capítol 2	11 días	lun 27/09/21	jue 07/10/21
11	Revisió PAC2	2 días	vie 08/10/21	dom 10/10/21
12	Entrega PAC2	0 días	dom 10/10/21	dom 10/10/21
13	**PAC 3: Producte projecte**	**54 días**	**lun 11/10/21**	**vie 03/12/21**
14	Correcció PAC2	2 días	lun 11/10/21	mar 12/10/21
15	Esborrany capítol 3	1 día	lun 11/10/21	lun 11/10/21
16	Implementació node físic	13 días	lun 11/10/21	sáb 23/10/21
17	Integració sensors	3 días	lun 11/10/21	mié 13/10/21
18	Integració mòdul comunicacions	5 días	jue 14/10/21	lun 18/10/21
19	Integració mòdul GPS	4 días	jue 14/10/21	dom 17/10/21
20	Integració sistema refrigerador	5 días	jue 14/10/21	lun 18/10/21
21	Proves funcionament node físic	5 días	mar 19/10/21	sáb 23/10/21
22	Comunicació node físic - servidor	5 días	dom 24/10/21	jue 28/10/21
23	Esborrany capítol 4	4 días	dom 24/10/21	mié 27/10/21
24	Creació dispositius simulats	6 días	vie 29/10/21	mié 03/11/21
25	Comunicació nodes simulats - servidor	6 días	jue 04/11/21	mar 09/11/21
26	Esborrany capítol 5	4 días	jue 04/11/21	dom 07/11/21
27	Implementació plataforma IoT	12 días	vie 29/10/21	mar 09/11/21
28	Connexió aplicació web	6 días	mié 10/11/21	lun 15/11/21
29	Creació dashboard	7 días	mar 16/11/21	lun 22/11/21
30	Creació panells	4 días	mar 16/11/21	vie 19/11/21
31	Enviament comandes	4 días	mar 23/11/21	vie 26/11/21
32	Proves entorn real	5 días	sáb 27/11/21	mié 01/12/21
33	Esborrany capítol 6,7,8	14 días	mar 16/11/21	lun 29/11/21
34	Esborrany capítol 9,10,11,12	2 días	mar 30/11/21	mié 01/12/21
35	Revisió PAC3	2 días	jue 02/12/21	vie 03/12/21
36	Entrega PAC3	0 días	vie 03/12/21	vie 03/12/21
37	**PAC 4: Memòria final**	**24 días**	**sáb 04/12/21**	**lun 27/12/21**
38	Correcció PAC3	2 días	sáb 04/12/21	dom 05/12/21
39	Recol. dades addicionals funcionament	10 días	sáb 04/12/21	lun 13/12/21
40	Redactat definitiu memòria	18 días	sáb 04/12/21	mar 21/12/21
41	Comentaris i neteja codi	1 día	mié 22/12/21	mié 22/12/21
42	Revisió i imprevistos	5 días	jue 23/12/21	lun 27/12/21
43	Entrega memòria final	0 días	lun 27/12/21	lun 27/12/21
44	**PAC 5: Presentació**	**9 días**	**mar 28/12/21**	**mié 05/01/22**
45	Elaboració presentació	7 días	mar 28/12/21	lun 03/01/22
46	Preparació presentació i defensa	2 días	mar 04/01/22	mié 05/01/22
47	Entrega presentació i defensa	0 días	mié 05/01/22	mié 05/01/22

1.5 Breu sumari de productes obtinguts

Amb la realització d'aquest treball s'obté una solució IoT integral de gestió de flotes pel transport frigorífic en la qual, mitjançant una comunicació bidireccional en temps real, es poden monitorar paràmetres com la posició i la temperatura del remolc dels vehicles des d'una plataforma IoT i a la vegada, enviar comandes de control als vehicles.

Com que es tracta del disseny i desenvolupament d'un nou producte es pretén realitzar tant un prototip per comprovar el funcionament del sistema en un entorn real, com un conjunt de vehicles simulats per verificar el funcionament de la plataforma creada per a la gestió IoT de la flota de vehicles.

Per tant, per una banda s'obté un node físic totalment funcional que capta les dades d'interès del vehicle com són la temperatura i la posició i les envia a la plataforma IoT en temps real; i, a la vegada, rep les ordres enviades des de la plataforma i actua sobre els elements instal·lats al vehicle com el generador de fred.

Per altra banda, s'obté una aplicació des de la qual es poden simular vehicles amb les mateixes característiques que les del node físic. Així, es poden simular els valors de temperatura i posició i enviar-los a la plataforma IoT.

I finalment, s'obté aquesta plataforma IoT que ha de permetre enregistrar, processar i presentar les dades recollides dels nodes (tant reals com simulats), monitorar les alarmes i les incidències, enviar les comandes als diferents vehicles de la flota i poder analitzar l'històric de dades.

1.6 Breu descripció dels altres capítols de la memòria

En el segon capítol d'aquest treball es presenta l'estat de l'art de la cadena de fred en el sector del transport, de les tecnologies i els sistemes de comunicació IoT utilitzats actualment en logística i de les plataformes IoT disponibles en el mercat.

En el tercer capítol es descriu i detalla l'arquitectura del sistema, des de que les dades són capturades pels sensors fins que són visualitzades a la plataforma IoT.

En el quart capítol es descriu la implementació del node físic tant a nivell de maquinari com de programari.

En el cinquè capítol es descriu la implementació dels nodes simulats, tant el mètode utilitzat per crear i simular els diferents dispositius IoT, com la configuració dels nodes i la comunicació amb la plataforma IoT.

En el sisè capítol es desenvolupa la plataforma Cloud, tant la part de gestió i presentació d'informació en temps real com l'anàlisi de la informació històrica registrada.

Finalment, en el setè capítol, s'extreuen les conclusions del treball.

2. Estat de l'art

2.1 La cadena de fred en el transport

La cadena de fred és el procés de subministrament de productes peribles a temperatura controlada, principalment en el sector alimentari i farmacèutic, des de la seva recollida o producció fins al seu consum. Aquests requereixen un estricte control de la cadena de fred per assegurar, en el cas dels aliments, que el producte és apte pel consum i que conserva la seva qualitat i frescor, i en el cas dels productes mèdics, que conserva les seves propietats i garanteix l'eficiència en el seu consum.

En el document [5] publicat per la OMS es realitza una comparativa de les principals solucions de monitorització de temperatura existents utilitzades en les operacions de transport, en aquest cas, de productes farmacèutics. Bàsicament es poden dividir entre dispositius electrònics i indicadors químics, en què els més utilitzats de cada conjunt són els *data loggers* i les etiquetes TTI (*Time-Temperature Indicator*), respectivament.

Els *data loggers* emmagatzemen la temperatura periòdicament a la seva memòria interna. Una vegada el producte arriba a destí, l'operari descarrega les dades a un ordinador o dispositiu mòbil mitjançant connexió USB, Bluetooth, RFID o NFC i comprova el registre de temperatures. Encara que es poden instal·lar a nivell de caixa, com que el seu cost és elevat, se solen instal·lar un o dos dispositius a nivell de cambra frigorífica.

A [6], per exemple, s'implementa una etiqueta passiva RFID per controlar les condicions dels productes farmacèutics durant el transport i determinar amb la seva lectura a l'arribada si els productes estan en condicions per al seu ús o emmagatzematge.

Empreses com DHL disposen d'un *data logger* de temperatura passiu que es pot adherir a paquets, palets o contenidors com es mostra a la Figura 1. En qualsevol moment, una simple lectura amb un telèfon intel·ligent o un lector especial equipat amb tecnologia NFC permet al destinatari verificar tot l'historial de temperatura i possibles oscil·lacions. Cada vegada que es realitza una lectura, totes les dades es carreguen i s'emmagatzemen de forma segura a un servidor per tal que el client pugui visualitzar-les a través d'una aplicació web o mòbil.

Figura 1. DHL SmartSensor adherit a paquets.
Extret de www.DHL.com

Les etiquetes TTI com les mostrades a la Figura 2, en canvi, estan formades per reactius químics sensibles als canvis de temperatura. Així, quan la cadena de fred es trenca, comença la reacció química i l'etiqueta canvia de color. Amb un simple cop d'ull, l'operari pot comprovar si el producte ha patit una ruptura de la cadena de fred. Encara que les seves petites dimensions i baix cost fa que es puguin instal·lar a nivell de palet, caixa o fins i tot a nivell de producte, són d'un únic ús obligant a l'empresa corresponent a renovar contínuament aquests dispositius.

Figura 2. Etiquetes TTI (Time-Temperature Indicator) de la marca WarmMark.
Extret de www.deltatraksouthamerica.com

Tot i que tots aquests dispositius faciliten en gran mesura el control de temperatura, la informació no s'obté fins que el producte arriba al destí de manera que si s'ha trencat la cadena de fred durant el transport, el producte estarà malmès i s'haurà de rebutjar.

Per superar aquestes limitacions, apareixen els sistemes de monitorització en temps real els quals utilitzen una tecnologia de comunicació (GPRS, 3G...) per transmetre les dades en temps real, així com alertes instantànies en cas de canvis de temperatura. D'aquesta manera es pot comprovar en qualsevol moment l'estat dels productes i així reaccionar amb més rapidesa en front a possibles canvis o ruptures de la cadena de fred.

Això suposa un canvi en la gestió de riscos de la cadena de fred, passant d'una gestió reactiva a una proactiva, és a dir, es passa de detectar i eliminar els productes malmesos de la cadena de subministrament a poder predir i eliminar els trencaments de la cadena de fred i les oscil·lacions de temperatura. Per a la gestió de la cadena de fred, són els primers passos cap a l'era de l'IoT.

A [7] es presenta el disseny d'un sistema de monitoratge de temperatura en temps real de la cadena de subministrament de llet crua, l'estructura del qual es pot veure a la Figura 3. S'utilitza un mòdul GPRS per enviar els valors de temperatura al servidor, que posteriorment poden ser visualitzats des de la pantalla d'una terminal.

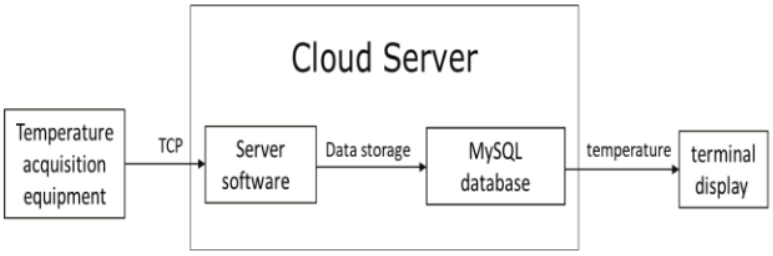


Figura 3. Estructura del sistema de monitoratge de [7]

Els sistemes de monitoratge en temps real que es comercialitzen actualment són bastant complerts, oferint una àmplia gama de sensors de mesura i multitud de serveis addicionals (eines de visualització, aplicacions mòbils...). Per aquesta raó, solen ser sistemes amb un elevat cost que solen anar dirigits a empreses amb un gran volum de negoci o per transports que requereixin un control molt exhaustiu de la temperatura (aliments, medicaments...).

Un exemple d'això són les inversions realitzades recentment per companyies com DHL o SEUR. SEUR disposa d'un servei de fred anomenat *SEUR Frio*, que permet el transport a una temperatura controlada d'entre 2 i 8 ºC a qualsevol part del territori nacional, en un termini de 24 hores.

L'operador logístic ha destinat aquest passat 2020 prop d'1 milió d'euros en el desenvolupament d'un sistema IoT per a la logística del fred. Amb la instal·lació de més de 1000 dispositius capaços de mesurar la temperatura amb tecnologia NB-IoT a totes les unitats de carrega de mercaderia (vehicles, contenidors, càmeres de fred...) i un sistema de gestió online es garanteix la traçabilitat de la temperatura des del moment de la recollida fins a la seva entrega.

2.2 L'IoT en el trasport

L'Internet de les Coses (*Internet of Things - IoT*) descriu la xarxa de dispositius i objectes que incorporen sensors, software i altres tecnologies amb l'objectiu de connectar i intercanviar dades amb altres dispositius i sistemes a través d'una xarxa (ja sigui privada o Internet). La finalitat última d'adquirir i intercanviar aquestes dades és processar-les i analitzar-les per obtenir informació valuosa i així prendre accions i decisions de negoci confiables. És aquí on també entra en joc el concepte de la Intel·ligència Artificial (IA) i particularment, l'aprenentatge automàtic (*Machine Learning* - ML) [8].

Junt amb els sistemes al núvol i el Big Data, l'IoT es posiciona com una de les 3 tecnologies amb més potencial i impacte per a les empreses logístiques i de transport. Segons el Baròmetre IoT [9] elaborat per Vodafone, el seu ús en aquest sector ha crescut del 19% al 27% en els últims 5 anys.

Els principals àmbits d'ús de les tecnologies IoT en el sector del transport i la logística són el control en temps real dels vehicles, de la càrrega i del propi conductor, com es mostra a la Figura 4. La combinació de tota aquesta informació és usada per millorar l'eficiència en la cadena de subministrament, reduir els temps, minimitzar costos i reduir les pèrdues de la càrrega.

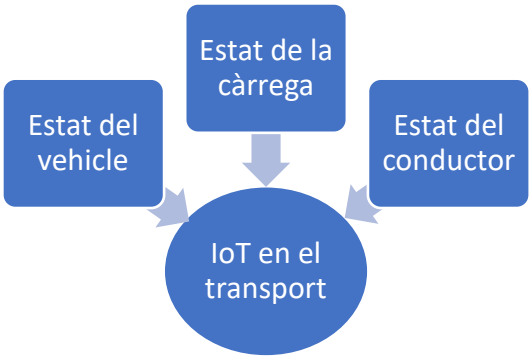


Figura 4. Principals àmbits d'ús de les tecnologies IoT en el transport.
Elaboració pròpia

En la **supervisió de l'estat del vehicle** s'inclou principalment el seguiment de la ubicació i la situació en temps real dels vehicles però també es pot incloure la velocitat del vehicle, les revolucions del motor, la pressió dels pneumàtics, el consum de combustible i l'obertura de portes.

Totes aquestes dades que es poden monitorar permeten realitzar diagnòstics en temps real sense haver de treure els vehicles de la carretera afavorint el manteniment proactiu de la flota i reduint així els costos de manteniment.

A [10] es dissenya i s'implementa un sistema de monitoratge del nivell de combustible d'una flota de vehicles. S'utilitza un Arduino per manipular les dades i enviar-les a través d'un mòdul GSM/GPRS a una base de dades al núvol. La informació és presentada en una aplicació web.

A [11] s'implementa un sistema en què s'utilitza el dispositiu OBDII per capturar alguns paràmetres del vehicle i codis de diagnòstic, juntament amb la posició del vehicle via GPS. Aquests són enviats mitjançant GPRS a un servidor d'Amazon juntament amb notificacions sobre paràmetres crítics.

En la **supervisió del conductor**, a través de la tecnologia IoT es pot detectar en temps real l'estat del conductor i els comportaments de conducció. La manera en què condueix el conductor repercuteix en aspectes com l'estalvi de combustible, la seguretat de la flota o la pròpia vida útil del vehicle. Així doncs, en aquest sentit, es poden analitzar paràmetres com el ralentí excessiu, les accelerades i frenades brusques, els excessos de velocitat, el temps de conducció o la falta d'ús del control de creuer.

Per altra banda, per detectar l'estat del conductor, la incorporació de una o diverses càmeres de vídeo, sensors portables (per controlar la freqüència cardíaca, la temperatura corporal, els nivells d'estrès) i l'ús de tecnologies avançades com la Visió per Computadora i el *Machine Learning* permeten detectar els casos de somnolència, les distraccions o l'incompliment de les normes de trànsit.

A [12] es presenta una solució en la qual un sistema integrat format per acceleròmetres, el sistema OBD-II i GPS permet avaluar l'estil de conducció segons la velocitat, l'acceleració, les frenades brusques, les revolucions del motor i el temps de conducció.

Actualment, empreses com Geotab tenen al mercat dispositius com el GO9, mostrat a la Figura 5, el qual permet obtenir dades sobre la ubicació, l'estat del vehicle, la detecció de col·lisions, el comportament al volant i l'estat de la bateria i del motor. Es connecta a l'OBD-II del vehicle i permet transmetre totes aquestes dades en temps real a través de connectivitat 2G, 3G i LTE.

Figura 5. Dispositiu Geotab GO9.
Extret de www.geotab.com

En la **monitorització de l'estat de la càrrega**, tal com el seu nom indica, es supervisa la ubicació de la mercaderia i l'estat de la càrrega en temps real. El fet de saber la posició de la mercaderia en tot moment afavoreix la seva visibilitat i permet ajustar les rutes i els horaris en funció del que està passant en aquell instant (retencions a la carretera, inclemències meteorològiques,...) proporcionant estimacions d'entrega més precises al client final.

Pel que fa a l'estat de la càrrega, alguns dels paràmetres que se solen mesurar amb els sensors corresponents són la temperatura, la humitat relativa, la lluminositat (per saber si s'ha manipulat la mercaderia), la concentració de gasos (com el CO_2), l'acceleració (per detectar cops i caigudes) i el pes de la càrrega.

A [9] es proposa un sistema de monitorització en temps real per a la cadena de fred de les vacunes en què s'utilitzen sensors de temperatura, humitat relativa, lluminositat i GPS. Aquestes dades s'envien mitjançant un mòdul GSM/GPRS a un servidor per posteriorment poder ser visualitzades en una aplicació mòbil.

Algunes empreses ja comercialitzen dispositius que ofereixen la supervisió i seguiment de qualsevol tipus d'actiu monitorant en temps real diversos paràmetres a través de diferents tipus de sensors i tecnologies de comunicació.

L'empresa NaBi, per exemple, ofereix un petit dispositiu IoT com el mostrat a la Figura 6, que pot fixar-se en qualsevol producte o actiu, per monitorar variables com la posició GPS, la temperatura, la vibració, l'acceleració i l'orientació. També permet rebre alertes en temps real si s'ha superat algun dels llindars establerts. Utilitza la xarxa NB-IoT per transmetre les dades als servidors d'Amazon i les dades es poden visualitzar mitjançant una aplicació mòbil, una aplicació web o es pot integrar amb sistemes externs.

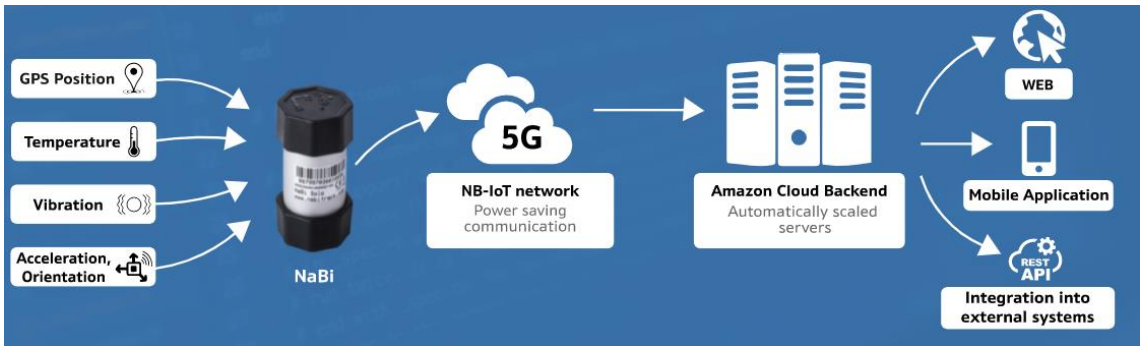


Figura 6. Funcionament del dispositiu NaBi Solo.
Extret de www.nabitrack.com

Així, amb la introducció de l'IoT en el sector del transport, els sistemes de transport comencen a "sentir" i pensar", la qual cosa porta al desenvolupament de tecnologies més avançades englobades en els anomenats Sistemes de Transport Intel·ligent (ITS) [13].

2.3 Tecnologies de comunicació en el transport

Un cop s'ha recopilat la informació d'estat (del vehicle, de la càrrega i/o del conductor) mitjançant els diferents sensors instal·lats i s'han preprocessat en el sistema integrat del vehicle, es requereix una interconnexió amb el centre logístic per transmetre totes aquestes dades en temps real.

Com que els objectes monitoritzats tenen un comportament dinàmic, serà necessari utilitzar una tecnologia de comunicació sense fils. L'ús final del sistema imposarà els requisits com la velocitat i capacitat de transmissió, l'estabilitat, la disponibilitat, la seguretat, el consum i, un dels diferenciadors més importants, l'abast. Així, segons l'abast que es requereixi per cobrir les necessitats de l'aplicació IoT destinada al sector del transport i la logística, les tecnologies de comunicació es poden classificar en els grups mostrats a la Figura 7.

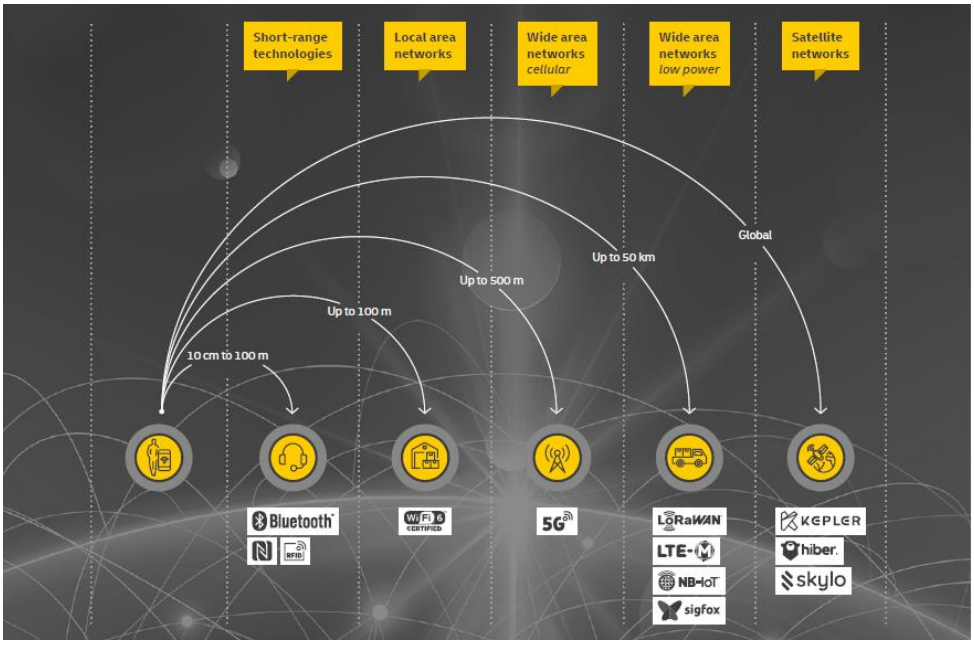


Figura 7. Abast de les principals tecnologies de comunicació sense fils.
Extret de [3].

A continuació, es descriuen els aspectes més destacats de cadascuna d'aquestes tecnologies de comunicació sense fils classificades segons el seu abast i aplicades al sector del transport i la logística.

- **Xarxes sense fils d'àrea personal (WPAN) i d'àrea local (WLAN)**

Són utilitzades sobretot en l'etapa d'emmagatzematge, transport i processament de la mercaderia. En aquests escenaris es transmeten les senyals d'estat del producte o de l'entorn des d'uns pocs centímetres a varis metres de distància. Algunes de les tecnologies englobades en aquests grups són RFID, Bluetooth, NFC, WiFi i ZigBee.

En tecnologies com RFID i NFC, la descàrrega de dades registrades s'ha de fer de forma manual una vegada s'arriba al punt d'entrega, la qual cosa impedeix la seva mesura en temps real. Això fa que, no només no es pugui tenir la possibilitat de realitzar un seguiment durant el transport, sinó que també l'encareix ja que en alguns casos els elements utilitzats no són reutilitzables.

Com a millora, moltes empreses impulsen la utilització de sensors sense fils connectats mitjançant Bluetooth, ZigBee o Wi-Fi que, encara que permeten connexions sense fils entre punts de mesura, necessiten d'encaminadors o portes d'enllaç (*gateways*) per enviar les dades a Internet, la qual cosa fa que els punts de mesura (nodes) no siguin compatibles amb una altra plataforma sense fins que la implementada, i a la vegada, depenen també d'aquests *gateways*. Tampoc proporcionen una cobertura suficient com per poder realitzar un seguiment de vehicles o monitorització d'actius en temps real.

A [14] es proposa un sistema de monitorització de la temperatura interior dels contenidors que viatgen dins un camió en què, utilitzant sensors Bluetooth, s'envien les dades al telèfon mòbil del conductor des d'on automàticament es pugen al servidor mitjançant la seva connexió a Internet, com es mostra a l'esquema de la Figura 8.

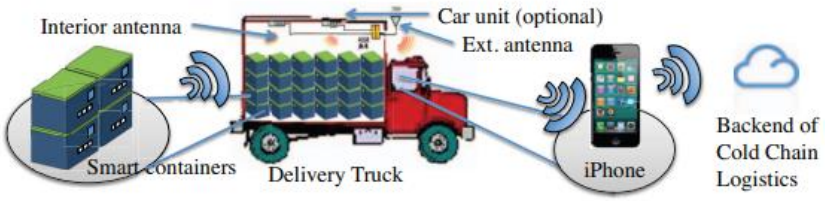


Figura 8. Subsistemes de la logística de la cadena de fred de [14]

- **Xarxes sense fils d'àrea estesa (WWAN)**

Proporcionen cobertura a nivell regional o nacional i s'utilitzen per connectar dispositius a distàncies d'uns quants centenars de metres a diversos centenars de quilòmetres. En aquest grup hi troben les xarxes tradicionals de telefonia, des del 2G fins l'actual 5G, però també les LPWAN (*Low Power Wide Area Networks*), com són NB-IoT, LTE-M, LoRaWAN, Sigfox i Weightless, les quals es tractaran de forma separada en el següent apartat.

Els productes IoT/M2M s'han desenvolupat convencionalment al voltat de les tecnologies de connectivitat basades en GSM. Encara que proporcionen una molt bona cobertura, per a la majoria d'aquests tipus d'aplicacions aquestes no són òptimes. Sovint no estan adaptades als missatges curts que s'envien en IoT, originant sobrecàrregues en les comunicacions, fent que siguin altament ineficients.

Les tecnologies GPRS, 3G i LTE proporcionen una cobertura suficient per a la majoria d'aplicacions relacionades amb el transport i la logística. Tot i això, cal considerar que l'augment de l'ample de banda implica una reducció de l'abast, fent que la cobertura no estigui tan uniformement estesa arreu del territori a mesura que s'escull una tecnologia superior. A més, els costos de maquinari del terminal, els costos de subscripció i el consum d'energia en molts casos són massa elevats per a una aplicació relacionada amb la gestió de flotes en què s'envien molt poques dades, llevat que hi intervinguin senyals d'àudio o vídeo.

Pel 5G, el Sector de Radiocomunicacions de la Unió Internacional de Telecomunicacions (ITU-R), encarregada d'establir els estàndards de les xarxes mòbils globals, ha definit tres escenaris fonamentals per a les futures xarxes 5G [15]. En un d'ells, l'anomenat *Massive Machine Type Communication* (mMTC), les connexions estaran dissenyades específicament per a dispositius IoT, posant especial èmfasi en el baix cost i el baix consum d'energia i la capacitat de les xarxes per suportar un gran nombre de dispositius.

- **Xarxes sense fils d'àrea estesa de baixa potència (LPWAN)**

Encara que el 5G ha estat la tecnologia sense fils de nova generació més difosa, no és l'únic candidat per connectar milions de dispositius que poden formar part de l'IoT. Recentment, tal i com es mostra a la Figura 9, han aparegut diverses alternatives dissenyades per satisfer les necessitats particulars dels dispositius IoT: baix cost, baix consum d'energia i alta fiabilitat.

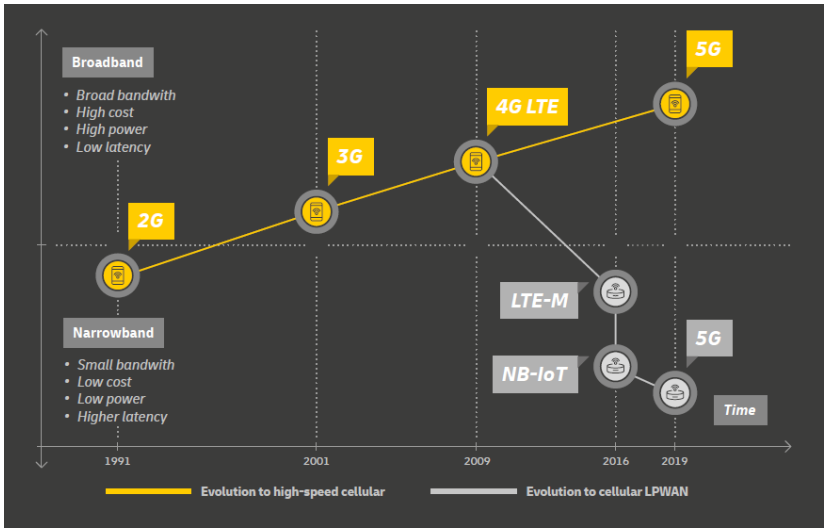


Figura 9. Evolució de les xarxes LPWAN en tecnologia mòbil.
Extret de [3]

Les tecnologies de xarxa d'àrea estesa de baixa potència (LPWAN) exploten el fet que els dispositius IoT intercanvien molta menys quantitat de dades que els ordinadors o els telèfons intel·ligents els quals estan habilitats per a multimèdia. Tal com es pot veure a la Figura 10, les LPWAN estan dissenyades per omplir un gran espai en l'ecosistema de comunicacions sense fils de propera generació, oferint la funcionalitat de les tecnologies de curt abast com Bluetooth i RFID, però amb un abast de quilòmetres com les tecnologies mòbils 4G i 5G les quals són molt més costoses.

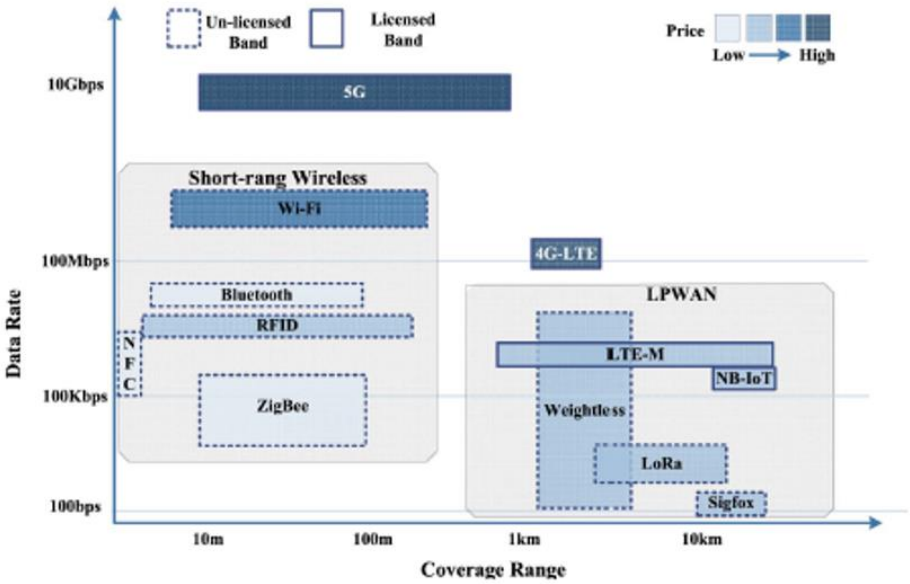


Figura 10. Principals tecnologies de comunicació sense fils per IoT
Extret de www.csit.carleton.ca/~fyu/Papers/09241736.pdf

A banda del seu baix cost, un altre avantatge de les tecnologies LPWAN és que ja tenen un ús comercial estès, cosa que permet als usuaris connectar-se amb la infraestructura existent. Si la cobertura encara no està disponible, es pot introduir de forma ràpida i econòmica amb una única estació base, la qual pot connectar una àrea de fins la mida d'una ciutat.

Les xarxes LPWAN es subdivideixen en dos tipus: les LPWAN derivades de les xarxes de telefonia actuals i les LPWAN independents.

Les LPWAN derivades de les xarxes de telefonia actuals estan dissenyades per utilitzar el mateix espectre amb llicència que actualment utilitza el 4G i per proporcionar serveis mitjançant la mateixa infraestructura. Els dos sistemes més implantats avui en dia són:

- **NB-IoT:** dissenyat per maximitzar la cobertura mitjançant la infraestructura d'estacions base existents, mantenint al mateix temps el consum d'energia i els costos de maquinari del dispositiu el més baix possible. Té una excel·lent penetració en interiors i sota terra. A diferència de LTE-M, no és tant adequat per mobilitat ja que el dispositiu ha de tornar a seleccionar la cel·la mentre es mou implicant més consum d'energia.
- **LTE-M:** amb la M de màquina, és una versió simplificada de la tecnologia LTE. Ofereix velocitats de dades i un grau de bidireccionalitat lleugerament més alt i una latència inferior a NB-IoT, la qual cosa permet la transmissió de veu i està preparada per dispositius en moviment, fent-lo adequat per aplicacions de transport i de seguiment de la cadena de subministrament. Per contra, requereix més energia per funcionar.

Pel que fa a les LPWAN independents, el potencial de creixement de la tecnologia LPWAN ha animat a altres organitzacions a entrar en aquest espai, oferint solucions que funcionen en parts de l'espectre sense llicència i funcionen amb la seva pròpia infraestructura. El llarg abast d'una única estació base LPWAN fa que la creació d'una xarxa des de zero sigui molt menys costosa que intentar replicar una xarxa mòbil convencional. Avui dia, dues tecnologies rivals han incidit significativament en aquesta àrea:

- **Sigfox:** ofereix una tecnologia de banda estreta mitjançant el seu propi protocol propietari. A través del desenvolupament del codi obert, l'empresa ha estat capaç d'establir els estàndards per a una xarxa 0G amb potència ultra baixa i ampli abast. Al ser un protocol privat, el seu servei és de pagament per subscripció.
- **LoRa:** és un estàndard obert però en el qual es limita el volum i la freqüència de dades, sent adequat només per transmissions de dades més aviat lentes i poc freqüents. Encara que avui en dia, només l'empresa Semtech fabrica els xips necessaris, un programa de certificació *LoRa Alliance* permet a qualsevol organització construir dispositius compatibles. Existeix un protocol de capa superior anomenat *LoraWAN*, basat en una xarxa amb gateway connectat als nodes.

Aquestes quatre tecnologies seran analitzades amb més detall en el tercer capítol d'aquest treball a l'hora d'analitzar els mòduls de comunicacions que es poden trobar actualment al mercat.

- **Xarxes sense fils d'àrea global de baixa potència (LPGAN)**

Empreses com Hiber, Iridium i Skylo estan començant a desplegar xarxes de petits satèl·lits geoestacionaris per proporcionar connectivitat global de baix cost en aplicacions d'IoT útils en àmbits com l'agricultura, la pesca, la logística o en cas de desastres naturals. A [16] es realitza un estudi sobre les tecnologies, estàndards i reptes oberts en les comunicacions IoT per satèl·lit.

A la Taula 1 es presenta un resum de les característiques tècniques més importants de les principals tecnologies de comunicació sense fils utilitzades en IoT.

	Freqüència	Rang	Velocitat	Consum d'energia	Nodes per gateway	Disponibilitat
RFID	125 kHz – 960 MHz	100 m	640 kBit/s	Molt baix	1	Infraestructura dedicada
NFC	13.56 MHz	10 cm	430 kBit/s	Baix	1	Telèfons intel·ligents
Bluetooth	2.4 GHz	350 m	1 MBit/s	Molt baix	Il·limitat	Telèfons intel·ligents i alguns gateways
WiFi 6	1, 2.4, 5 i 6 GHz	50 m	10 GBit/s	Mig	250	Telèfons intel·ligents i noves infraestructures
4G-LTE	410 – 5900 MHz	10 km	1 Gbit/s	Baix	600	Desenvolupament públic
5G	2.5 GHz – 49 GHz	500 m	50 GBit/s	Molt alt	1 milió	Desenvolupament públic limitat
LTE-M	500 – 5900 MHz	100 km	375 kBit/s	Baix	No disponible	Totes les xarxes de telefonia (segons disponibilitat)
NB-IoT	698 – 2170 MHz	40 km	250 kBit/s	Baix	Ca. 20000	Totes les xarxes de telefonia (segons disponibilitat)
LoRaWAN	490 MHz 868 MHz 915 MHz	50 km	300 Bit/s	Molt baix	1 milió	157 països
Sigfox	433 MHz 915 Mhz	50 km	100 Bit/s	Molt baix	1 milió	70 països
Satèl·lit	400 MHz	Global	144 Bit/s	Mitja	No aplicable	Global

Taula 1. Característiques tècniques de les principals tecnologies IoT sense fils

2.4 Protocols de comunicació en IoT

Un cop el sistema integrat del vehicle està connectat a una xarxa de comunicació sense fils, és necessari un protocol de comunicació a nivell de capa d'aplicació per a que pugui intercanviar informació amb el centre logístic en temps real.

Tenint en compte que els dispositius IoT sovint tenen una capacitat de processament limitada, un baix ample de banda, una capacitat de memòria reduïda i una autonomia limitada (si està alimentat per bateries), serà molt important l'eficiència de la comunicació la qual vindrà determinada, en gran part, per l'elecció del protocol de comunicació. Aquest dependrà de factors com la naturalesa del sistema IoT, la compatibilitat amb els dispositius i els requisits específics com el consum d'energia, l'amplada de banda, la seguretat o la privacitat.

Existeixen multitud de protocols IoT disponibles tant a nivell de consumidor com a nivell industrial: MQTT, CoAP, AMQP, XMPP, HTTP, OPC UA... els quals tenen els seus avantatges i inconvenients en funció de les aplicacions IoT. A articles com [17] i [18] es descriuen i s'analitzen les seves característiques, així com els principals problemes de rendiment incloent la latència, el consum d'energia o la capacitat de la xarxa.

Aquest apartat se centrarà en avaluar els tres protocols que, segons la comunitat global *oneM2M*, encarregada de desenvolupar estàndards IoT, són actualment els principals protocols per a les comunicacions M2M/IoT. A més, com es veurà més endavant, són els que es poden trobar més estesos en les principals plaques de desenvolupament del mercat i els que ofereixen compatibilitat amb les principals plataformes IoT.

Aquests protocols són HTTP, MQTT i CoAP els quals es descriuen i es comparen en més detall a continuació.

- **HTTP (Hypertext Transfer Protocol)**

HTTP es pot considerar el principal protocol de comunicació al ser un dels més utilitzats globalment a nivell web, sobretot per a la comunicació en clients basats en navegadors. Funciona sobre TCP i està basat en el model client/servidor (o petició/resposta). És un protocol robust i que pot arribar a oferir un alt grau de seguretat de manera que pot ser útil en certes aplicacions basades en IoT.

Quant a seguretat utilitza TLS/SSL i quant a fiabilitat no es defineix de manera explícita QoS, requerint suport addicional per a això.

No obstant, es requereixen dispositius amb un poder de computació elevat i un canal de comunicació amb un bon ample de banda ja que HTTP és un protocol relativament pesat, que consumeix molta energia i ineficient per escenaris de comunicació d'1 a n dispositius (es necessita fer un POST a cada client).

Per altra banda, tampoc està optimitzat per a una comunicació bidireccional ja que si el servidor ha d'enviar dades al client, és el client el que ha de preguntar periòdicament al servidor si hi ha noves dades disponibles per a ell (*polling*). Això, juntament amb la seva elevada latència fa que no sigui recomanable utilitzar-lo en aplicacions de temps real.

Aquestes limitacions com a protocol de comunicació IoT ha provocat l'aparició d'altres protocols més optimitzats per a ser utilitzats específicament amb xarxes i nodes de recursos limitats, com és el cas de MQTT i CoAP.

- **MQTT (Message Queuing Telemetry Transport)**

MQTT és un protocol lleuger, escalable i fàcil d'implantar. Funciona sobre TCP i està basat en el model Publicació/Subscripció (Pub/Sub). Com es pot veure a la Figura 11, els dispositius publiquen informació sobre un tema o tòpic a un servidor central anomenat *broker* i aquest la reenvia a aquells clients que s'hagin subscrit al mateix.

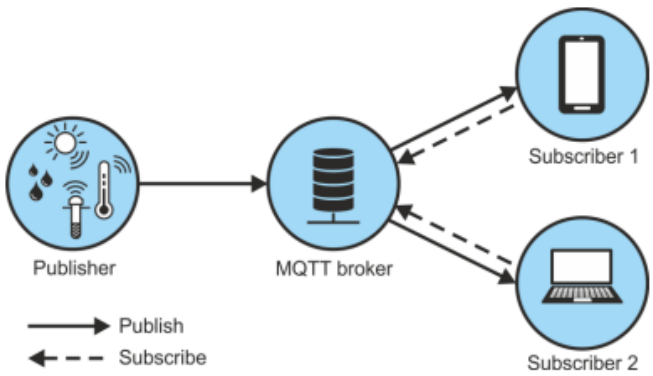


Figura 11. Arquitectura del protocol MQTT
Extret de e-tarjome.com/storage/panel/fileuploads/2019-12-16/1576500942_gh100.pdf

Això permet una comunicació de n a n dispositius on cada client pot produir i consumir dades publicant missatges i subscrivint-se a tòpics a través del broker. El gran avantatge d'aquest tipus de comunicació és que els dispositius IoT poden enviar dades del sensor i, alhora, rebre informació de configuració i comandes de control.

Així doncs, a diferència del protocol HTTP, els clients no han d'enviar una petició periòdicament per saber si hi ha noves dades disponibles per a ell, sinó que si hi ha dades noves, el broker els hi enviarà automàticament (mètode *push*).

És un protocol ideal per a petits dispositius amb pocs recursos i que requereixen un ús eficient de l'amplada de banda i de la bateria. És més ràpid, té menys sobrecàrrega i consumeix menys energia que l'HTTP. A més, es poden afegir nous dispositius sense tocar la infraestructura existent ja que com que aquests només es comuniquen amb el broker, no cal que siguin compatibles amb la resta de dispositius.

No obstant, el seu funcionament sobre TCP i la gestió de noms de tòpics molt llargs, pot suposar un problema per alguns dispositius de recursos molt limitats. Això es pot solucionar utilitzant la variant MQTT-SN (*MQTT for Sensor Networks*) que utilitza UDP i suporta indexació de noms de tòpics [19].

Quant a seguretat, existeixen 3 maneres en què el broker pot verificar la identitat d'un client MQTT:

- **ID del client:** tots els clients han de proporcionar un identificador de client que s'utilitza per enllaçar el tòpic amb el client i amb la connexió TCP quan un client es subscriu a un tòpic.
- **Noms d'usuaris i contrasenyes:** abans d'establir la connexió, el broker pot requerir nom d'usuari i contrasenya al client. Cal tenir en compte però que aquestes dades no estan encriptades i per tant, es transmeten en text pla.
- **Xifratge TLS/SSL:** permet xifrar les dades que es transmeten i autenticar els dispositius al servidor mitjançant l'ús de certificats. No obstant, fa augmentar la latència a l'haver d'intercanviar més dades; els certificats poden ser fitxers grans originant problemes de memòria al dispositiu i fins i tot, pot ser que no estigui suportat pel client, sobretot si es tracta de dispositius senzills i econòmics.

Quant a la fiabilitat, MQTT utilitza 3 nivells de Qualitat de Servei (QoS) per a cada connexió amb el broker:

- **Nivell 0:** el missatge s'envia una vegada i ni client ni broker prenen cap mesura addicional per confirmar el lliurament.
- **Nivell 1:** el missatge es pot reenviar múltiples vegades fins que es rebi la confirmació.
- **Nivell 2:** emissor i receptor estableixen un handshake per assegurar que només una còpia del missatge és rebuda.

Quant al broker, hi ha dues opcions principals on allotjar el broker MQTT:

- **Servidor instal·lat localment:** existeixen diversos brokers MQTT gratuïts i de codi obert com Mosquito, Mosca, emqttd...
- **Servidor basat en el núvol:** empreses com Google, Amazon, Microsoft i IBM proporcionen servidors/brokers MQTT al núvol.

- **CoAP (Constrained Application Protocol)**

CoAP també és un protocol lleuger adequat per utilitzar en xarxes i nodes amb recursos limitats. Funciona sobre UDP i presenta una arquitectura client/servidor com es msotra a la Figura 12. CoAP i HTTP tenen moltes funcions similars amb la diferència que CoAP està optimitzat per a IoT.

Per tant, funciona basant-se en l'intercanvi de missatges asíncrons entre dos nodes, un actuant com a client que envia peticions a l'altre, el servidor, el qual atendrà aquesta petició. A diferència de MQTT, CoAP utilitza l'Identificador de Recursos Universal (URI) en lloc del tòpics. De la mateixa manera que HTTP, CoAP es basa en el model REST: els servidors fan que els recursos estiguin disponibles en una URL, i els clients accedeixen a aquests recursos utilitzant mètodes com GET, PUT, POST i DELETE. CoAP s'integra amb XML, JSON i CBOR.

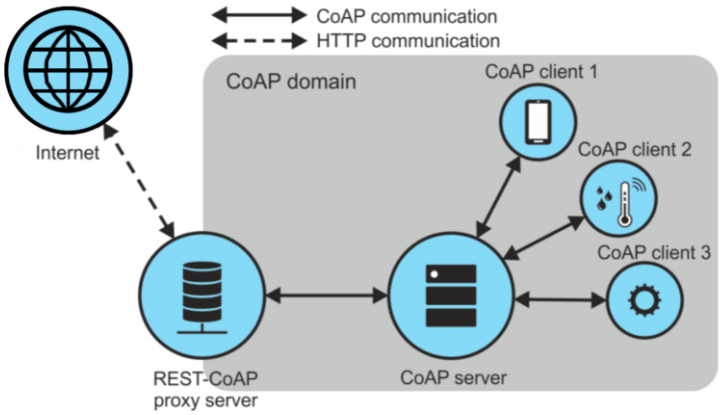


Figura 12. Arquitectura del protocol CoAP
Extret de e-tarjome.com/storage/panel/fileuploads/2019-12-16/1576500942_gh100.pdf

Quant a seguretat, CoAP, de la mateixa manera que HTTP i MQTT, és un protocol no xifrat per defecte, la qual cosa significa que per assegurar la comunicació també es necessita un protocol addicional. En aquest cas, CoAP utilitza DTLS (*Datagram Transport Layer Security*) per assegurar i protegir la informació.

Quant a fiabilitat, encara que CoAP no proporciona un model de QoS explícitament, sí que utilitza un sistema semblant a través dels missatges confirmables (CON) i no confirmables (NON). Els primers són aquells en què el servidor acusa la seva recepció amb un paquet ACK com a resposta, mentre que els segons són aquells en què el servidor no informa de la seva recepció.

La seva adopció al mercat no està tan estesa com HTTP o MQTT, la qual cosa es poden veure limitades les opcions de software i hardware.

A la Taula 2 es presenta un resum de les principals característiques dels tres protocols descrits anteriorment.

	MQTT	CoAp	HTTP
Arquitectura	Client/Broker	Client/Servidor o Client/Broker	Client/Servidor
Paradigma	Publicació/Subscripció	Publicació/Subscripció Petició-Resposta	Petició-Resposta
Mida capçalera	2 bytes	4 bytes	Indefinit
Mida del missatge	Petit i indefinit	Petit i indefinit	Gran i indefinit
Nodes de comunicació	M:N	1:1	1:1
Capa de transport	TCP (MQTT-SN: UDP)	UDP	TCP
Seguretat	TLS/SSL	DTLS, IPSec	TLS/SSL
Port per defecte	1883/8883 (TLS/SSL)	5683 (UDP) / 5684 (DTLS)	80/443 (TLS/SSL)
Fiabilitat	3 nivells de QoS: QoS 0: Com a màxim 1 cop QoS 1: Almenys 1 cop QoS 2: Exactament 1 cop	Missatges CON Missatges NON	Limitat (via TCP)
Format de codificació	Binari	Binari	Text
Model de llicència	Codi obert	Codi obert	Lliure
Consum d'energia	Baix	Molt baix	Alt
Altres	Fàcil d'implementar Complex per afegir extensions	Poques llibreries existents Baixa latència	Moltes llibreries

Taula 2. Característiques dels principals protocols de comunicació IoT

2.5 Sistemes de computació encastats per a IoT

Els sistemes de computació encastats per a IoT existeixen en una àmplia varietat de plataformes hardware. Des de petits microcontroladors de 8 bits fins a les CPU de x86 més recents que es troben en equips de sobretaula. A l'hora d'escollir el hardware per a un dispositiu IoT intervenen moltes variables, entre elles són:

- **Cost:** se solen utilitzar dispositius més petits i barats quan el producte final es produeix en massa. L'inconvenient és que el desenvolupament del dispositiu pot resultar més car degut a que es tracta de dispositius altament restringits.
- **Consum:** si el dispositiu utilitza bateries i no està connectat a la xarxa elèctrica serà un aspecte a tenir en compte. Els microcontroladors solen estar dissenyats per escenaris de menor consum i poden ser una millor opció per ampliar la duració de la bateria.
- **Accés a la xarxa:** hi ha moltes maneres de connectar un dispositiu a un servei al núvol. Ethernet, Wi-Fi i telefonia mòbil són algunes de les opcions disponibles. El tipus de connexió que s'esculli dependrà del lloc en el qual s'implementi el dispositiu i de com s'utilitzi. Per exemple, la xarxa de telefonia mòbil pot ser una opció atractiva gràcies a l'elevada cobertura; no obstant, en els casos dels dispositius de tràfic elevat, pot resultar cara. Ethernet per cable proporciona uns costos de dades més econòmics, però té l'inconvenient de ser menys portàtil.
- **Entrades i sortides:** les entrades i sortides disponibles en els dispositius afecten directament a la capacitat operativa d'aquests. Un microcontrolador normalment tindrà moltes funcions d'E/S integrades directament al xip i proporcionarà una àmplia gamma de sensors per connectar-se directament.

Els dispositius IoT poden dividir-se en dos àmplies categories: microcontroladors (MCU) i microprocessadors (MPU).

Els MCU són menys costosos i més fàcils d'utilitzar que els MPU. Un MCU conté moltes de les funcions com, per exemple, la memòria, les interfícies i les E/S dins del propi xip. Un MPU obtindrà aquesta funcionalitat dels components dels xips compatibles. Un MCU sol utilitzar un sistema operatiu en temps real (RTOS) o funcionar sense sistema operatiu, i proporcionar respostes en temps real i reaccions molt deterministes a esdeveniments externs.

MCU és més adequat per programes petits i mitjans amb càlculs relativament simples, que generalment s'utilitzen per a la gestió i control de hardware. Per tant, generalment, no requereixen de RAM i ROM externes, i estan integrades en el xip, també anomenat microordinador d'un sol xip.

Els MPU solen executar un sistema operatiu d'ús general, com Windows, Linux o MacOSX, que proporcionen una resposta en temps real no determinista. Normalment, no hi ha cap garantia de quan es completarà una tasca. Solen ser més adequats per programes a gran escala que requereixen càlculs complexes, que generalment requereixen de RAM i ROM externes.

Per al desenvolupament d'aquest projecte, s'ha decidit analitzar 3 de les plataformes hardware més utilitzades actualment, mostrades a la Figura 13: Arduino, STM32 (NUCLEO) i Raspberry Pi.

Figura 13. Dispositius IoT. a) Arduino UNO. b) Nucleo STM32. c) Raspberry Pi 4.
Extret de www.arduino.cc, www.st.com i www.raspberrypi.org

Arduino és una de les plaques de prototipatge més populars del mercat. No obstant, la plataforma no ha evolucionat gairebé gens en el temps i en termes de hardware ha quedat una mica desfasada. Comparant amb les plaques NUCLEO, aquestes tenen un rendiment, qualitat, entrades/sortides i opcions de depuració molt superiors. No obstant, la placa NUCLEO no disposa de memòria EEPROM que permeti emmagatzemar variables permanents en cas de reiniciar el sistema. En canvi, Arduino té una EEPROM incorporada amb el microcontrolador ATMEL.

En termes de rendiment, NUCLEO pot processar algorismes més complexes gràcies al coma flotant i també el compilador C s'escriurà amb moltes menys instruccions, permetent una execució més ràpida i una eficiència significativa dins dels programes.

STM32 s'utilitza principalment per recopilar dades de sensors, analitzar les dades, processar-les i controlar el dispositiu de sortida per completar les funcions corresponents que poden ser des de controlar un motor, una pantalla LCD... Aquest tipus de productes generalment no requereixen càlculs complexes i executen un sistema operatiu RTOS o es programen directament.

Per tant, un microordinador d'un sol xip és més adequat per a l'adquisició i el control de dades. El seu avantatge és que té un alt rendiment en temps real. No és necessari esperar a que s'iniciï el sistema operatiu. El programa es pot iniciar immediatament després que s'hagi encès.

La majoria de models de STM32 pertanyen a la sèrie de microcomputadores d'un sol xip Cortex-M, i la Raspberry Pi és generalment de la sèrie de processadors Cortex-A.

La diferència principal és que les versions Cortex-M (M0, M3, M4...) són les versions de menys rendiment. Es tracta d'un microcontrolador que generalment té un baix cost i consum d'energia però té un rendiment limitat. Les principals àrees d'aplicació inclouen cases intel·ligents, drons, dispositius portàtils intel·ligents i electrònica automotriu (airbags...).

Les versions Cortex-A són les de més alt rendiment i s'utilitza en productes de gamma alta com tauletes i telèfons mòbils.

Per altra banda, també existeixen les versions Cortex-R (R4, R5, R7...) que estarien en un punt intermedi amb un rendiment mig i que s'utilitzen per aplicacions industrials d'alt rendiment en temps real, com sistemes de frenat d'automòbils, control de motors, impressores, controladores de disc dur...

El potent rendiment de Raspberry Pi fa que sigui més adequat per a la informàtica, com l'adquisició, el processament, l'aprenentatge profund i el reconeixement d'imatges. El rendiment en temps real d'aquest sistema Linux no és tan alt com el d'un microordinador d'un sol xip. Pel desenvolupament de productes el seu cost és massa elevat.

A la Taula 3, es presenta una comparativa de les característiques més importants de cadascun dels dispositius IoT analitzats en aquest projecte.

	Arduino Uno	Nucleo F401	Raspberry Pi 4
Tipus de CPU	Microcontrolador	Microcontrolador	Microprocessador
SoC	ATMega 328 8-bit	STM32F401 32 bit	BCM2711
Familia	AVR	ARM Cortex M4	Cortex A??
Sistema operatiu	-	Instalable (FreeRTOS...)	Linux (Raspbian)
CPU	16 MHz	84 MHz	Quad Cortex-A72 @ 1.5 GHz
RAM	2KB	96 Kb	1GB, 2 GB o 4 GB
Memòria Flash	32 KB	512 Kb	Depèn targeta SD
Emmagatzematge	EEPROM 1 KB	-	MicroSD
Entrades analògiques	6	16	-
Pins digitals I/O	14	47	40
Connectivitat	Cap de sèrie, es poden afegir via Shields	Cap de sèrie, es poden afegir via periférics	USB, HDMI, Ethernet, Wi-fi, Bluetooth...
Preu	22 €	14 €	Des de 37 €

Taula 3. Característiques de les principals plataformes hardware per a IoT

2.6 Plataformes IoT

Un cop les dades dels dispositius IoT arriben al centre logístic mitjançant una de les tecnologies de comunicació descrites anteriorment, es requereix d'una plataforma que permeti recollir les dades, emmagatzemar-les i processar-les per tal que l'usuari final pugui visualitzar la informació i els resultats mitjançant una interfície gràfica o *dashboard*. Moltes d'aquestes plataformes també permeten enviar comandes als dispositius, processar alertes, gestionar les versions de l'aplicació, administrar els drets d'accés dels usuaris, analitzar les dades i generar informes, entre moltes altres funcions.

També es coneixen com a *IoT Application Enablement Platform* (AEP) o plataformes IoT com a servei (PaaS), és a dir, que es poden crear aplicacions i serveis propis mitjançant les seves funcions IoT integrades.

Una de les consideracions inicials a tenir en compte a l'hora de posar en marxa una plataforma IoT és decidir si crear-la des de zero o utilitzar una de les plataformes existents basades en el núvol. Alguns dels factors a tenir en compte són els següents:

- **Termini de llançament:** si es disposa d'un termini curt, serà més apropiat utilitzar una plataforma existent pel fet que s'aconsegueix reduir el temps de desenvolupament.
- **Personalització:** caldrà avaluar si la plataforma basada en el núvol pot proporcionar una solució a les necessitats i requisits del projecte.
- **Maquinari:** per executar una plataforma IoT pròpia serà necessari adquirir i mantenir el maquinari adequat, sent necessari valorar el cost de la inversió.
- **Capacitat i recursos interns:** si es decideix crear la plataforma IoT caldrà valorar si es tenen els mitjans necessaris per desplegar i mantenir els servidors, implementar i mantenir les bases de dades, dissenyar la interfície gràfica, complir amb les condicions de seguretat, entre d'altres.
- **Dades:** caldrà valorar si és necessari tenir el control total de les dades amb una plataforma pròpia o es pot cedir la informació i el control a un proveïdor extern.
- **Cost total:** tenint en compte el pressupost total del projecte caldrà valorar si el pagament per ús d'una plataforma basada en el núvol en què tots els costos ja estan inclosos és més econòmic que crear una plataforma pròpia des de zero amb totes les despeses auxiliars que comporta.

Tant si es crea des de zero com si s'utilitza una de les plataformes existents, caldrà valorar aspectes com la fiabilitat, la personalització, l'escalabilitat, els protocols acceptats, la seguretat, el suport i el preu.

Respecte les plataformes IoT basades en el núvol, les que estan liderant actualment el mercat són Amazon Web Services IoT, Google Cloud IoT i Microsoft Azure IoT. A continuació es descriuen les principals característiques que destaquen de cadascuna d'elles.

- **AWS IoT**

AWS IoT Core és el producte principal de la suite de AWS IoT. Gestiona l'autenticació, la connexió i la comunicació dels dispositius amb els serveis AWS. Suporta els protocols MQTT, HTTP i Websockets. Permet administrar fins a mil milions de dispositius amb una identitat única. Gràcies a l'autenticació i xifratge proporcionat en tots els punts de connexió, l'IoT Core i els dispositius mai intercanvien dades no verificades.

Mitjançant el *Rules Engine*, els missatges són encaminats cap a un dispositiu o servei AWS al núvol com *AWS Lambda* (plataforma sense servidor), *Amazon Kinesis* (solució per processar Big Data en temps real) o *Amazon S3* (servei d'emmagatzematge).

Una altra característica de l'IoT Core és el *Device Shadow*, que emmagatzema l'estat actual o desitjat de cada dispositiu. Així, si aquest està fora de línia o ocupat, les aplicacions al núvol encara poden canviar-ne la configuració o enviar-li ordres. Quan torni a estar en línia, sincronitza el seu estat final amb les actualitzacions enviades.

Ofereix SDK per a aplicacions d'Android i iOS, així com per a C, C++, JavaScript i Phyton. Amazon agilitza el desenvolupament proporcionant una gran col·lecció de plantilles juntament amb una eina visual *Drag&Drop* anomenada *IoT Things Graph* que simplifica la creació de fluxos de treball entre components de l'IoT.

Amazon facilita la computació Edge proporcionant un sistema operatiu en temps real per a microcontroladors anomenat *FreeRTOS*, el qual permet connectar els dispositius de forma segura a IoT Core o a altres serveis AWS.

- **Google Cloud IoT**

Google Cloud IoT Core és el component principal de la suite i està formada per dos mòduls: el Device Manager permet configurar, autenticar i controlar els dispositius de forma remota; i el Protocol Bridge que és el responsable de la connectivitat i que permet treballar amb els protocols MQTT i HTTP. Publica fluxos de dades al servei Cloud Pub/Sub que permet combinar missatges de diferents fonts en un sol sistema.

Des de Cloud Pub/Sub, les dades es reenvien a altres serveis al núvol de Google com *Cloud DataFlow* (preprocessament de dades en temps real), *Cloud Bigtable* (ingerir i emmagatzemar grans volums dades) i *BigQuery* (anàlisi de dades en temps real, creació i entrenament dels models de aprenentatge automàtic).

- **Microsoft Azure IoT**

IoT Hub és el producte central de la suite que permet la connectivitat, gestió i comunicació de dispositius. Es presenta en dos nivells, bàsic i estàndard, que es diferencien pel nombre de funcions habilitades.

El nivell bàsic permet: la comunicació del dispositiu al núvol, autenticació de dispositius, compatibilitat amb els protocols HTTP, MQTT, AMQP i Websockets i supervisió i diagnòstic de dispositius. El nivell estàndard afegeix la comunicació del núvol al dispositiu, gestió de dispositius, emmagatzematge d'informació sobre les propietats actuals i desitjades dels dispositius i *l'IoT Edge* per crear mòduls i desplegar-los als nodes de xarxa.

IoT Central és un SaaS (Software as a Service) escalable que permet un disseny àgil de software IoT amb funcions de seguretat integrades. La plataforma inclou funcions integrades de supervisió i gestió de dispositius per connectar, reconfigurar i actualitzar dispositius. El mòdul inclou nombroses plantilles d'aplicacions per a diferents indústries per accelerar el seu desenvolupament.

Microsoft Azure IoT també ofereix *Azure Device Twins* (de funcionament similar a *Device Shadow* de AWS IoT), *Azure Sphere* (solució de seguretat per protegir dispositius IoT, sistemes operatius i serveis al núvol) i *Time Series Insights* (extreu dades del IoT Hub per analitzar-les, detectar tendències, identificar anomalies i presentar els resultats de forma visual).

Com es pot veure, les tres plataformes IoT dels grans líders tecnològics tenen el mateix propòsit i funcionalitats similars. Totes elles presenten una alta escalabilitat que s'adapta a les necessitats de qualsevol empresa (des d'empreses emergents a empreses amb milions de dispositius), una seguretat integrada per a cada capa del sistema IoT i assistència tècnica i documentació detallada sobre els seus productes.

Per tant, el que pot fer decantar cap a una opció o una altra finalment són aspectes com la personalització o plantilles predefinides que disposi, els protocols acceptats i el preu.

Quant al preu, en totes les plataformes s'ofereix una solució bàsica que sol incloure un conjunt limitat de funcions. Tots els serveis que s'utilitzen a banda de les funcions bàsiques comporten despeses addicionals. Totes les plataformes funcionen amb un model de pagament per ús en què el preu total depèn del volum d'ús, del nombre de missatges o megabytes intercanviats, del nombre de dispositius connectats, del nombre d'accions executades...

Per altra banda, també permeten provar la plataforma mitjançant els seus plans gratuïts per tal de poder comprovar si compleix amb els requisits abans de començar a invertir en serveis concrets. A la Taula 4 es presenta una comparació de les característiques de les plataformes IoT descrites anteriorment.

	Amazon Web Services IoT	Microsoft Azure IoT	Google Cloud IoT
IoT SaaS	-	Azure IoT Central	Android Things Console (data tancament: gener 2022)
IoT PaaS	AWS IoT Core	Azure IoT Suite	Cloud IoT Core
Serveis IoT clau	AWS IoT Core Greengrass Core	Azure IoT Hub Azure IoT Edge	Cloud IoT Core
Protocols de comunicació	HTTP MQTT Websockets	HTTP MQTT AMQP Websockets CoAP (via Azure IoT Protocol Gateway Framework)	HTTP MQTT (via Cloud Pub/Sub) gRPC
OS Embegut	Amazon FreeRTOS	Windows 10 IoT	Android Things (data tancament: gener 2022)
Seguretat dels dispositius	TLS Certificat X.509	TLS Certificat X.509	TLS Certificat X.509 JWT
SDKs	AWS IoT Device SDK (C++, Phyton, JavaScript, Java, Android, iOS)	Azure IoT Device SDK (C, C#, Java, Node.js, Python)	Cloud IoT Device SDK (C)
Simulació de dispositius	AWS IoT Device Simulator	Azure IoT Device Simulator	-
Visualització de dades	Amazon QuickSight	Power BI	Google Data Studio
Preus	Per dispositiu + tràfic + emmagatzematge + serveis	Per dispositiu + tràfic + emmagatzematge + serveis	Per dispositiu + tràfic + emmagatzematge + serveis
Pla de prova	12 mesos, 300$ de prova, es requereix targeta de crèdit	1 mes, 200$ de prova, es requereix targeta de crèdit	300$ de prova, es requereix targeta de crèdit
Nivell gratuït	Sí (amb límit d'ús per mes)	Sí (amb límit d'ús per mes)	Sí (amb límit d'ús per mes)
Principals casos d'ús	Smart city Smart home Agricultura	Salut Venda la detall Fabricació	Energia Smart parking Transport i logística

Taula 4. Característiques de les plataformes IoT dels grans líders tecnològics

Cal tenir present que aquestes tres plataformes no són les úniques possibilitats que es poden trobar al mercat. Existeixen solucions que estan específicament dissenyades per a IoT amb opcions de personalització més avançades, preus i plans gratuïts més competitius, amb millor interoperabilitat i amb grans comunitats al seu darrere oferint suport. A [20] es realitza una classificació i comparació de les plataformes IoT més populars basada en les funcions bàsiques, de detecció, de comunicació i de desenvolupament d'aplicacions.

A la Taula 5 es proposen i comparen les característiques principals de tres d'aquestes solucions, en aquest cas de codi obert, que també es consideren perfectament vàlides per complir amb les necessitats i els requeriments d'aquest treball.

	Kaa IoT Cloud	ThingsBoard	Thinger.io
Protocols de comunicació	HTTP MQTT CoAP Websockets	HTTP MQTT CoAP	HTTP MQTT Websockets
Seguretat dels dispositius	TLS Certificat X.509	TLS Certificat X.509	TLS/SSL
Simulació de dispositius	Sí	Sí	No
Visualització de dades	Dashboard, gràfics, mapes, taules... en temps real	Dashboard, gràfics, mapes, taules... en temps real	Dashboard, gràfics, mapes, taules... en temps real
Preus	Segons número de dispositius	Segons número de dispositius i missatges enviats	Segons número de dispositius, plugins, cloud privada, dominis...
Pla de prova	Sí, fins a 5 dispositius, temps indefinit, no es requereix targeta de crèdit	Sí, dispositius il·limitats, temps indefinit, no es requereix targeta de crèdit, alguns serveis limitats	Sí, fins a 2 dispositius per temps indefinit, no es requereix targeta de crèdit, sense suport MQTT
Principals casos d'ús	Agricultura Smart City Smart Energy Transport i logística	Agricultura Smart Energy Smart Metering Transport i logística	Agricultura IoT Industrial Smart Cities Transport i logística

Taula 5. Característiques de tres plataformes IoT de codi obert

Totes aquestes aplicacions son adequades per configurar solucions d'IoT típiques, però mentre que aquestes últimes estan específicament dissenyades per a IoT, les dels tres grans líders tecnològics (AWS, Microsoft i Google) són plataformes al núvol més orientades a l'ecosistema en general.

Per aquest motiu, plataformes com Kaa IoT Cloud requereixen menys experiència tècnica per part de l'usuari ja que estan dissenyades per proporcionar totes les funcions necessàries en un sol lloc i sota una única interfície d'usuari, així com per tractar de manera predeterminada un flux IoT d'extrem a extrem, des de la recopilació fins a la visualització. A diferència d'això, les de AWS, Microsoft i Google requereixen configurar tot un ecosistema complert, presenten fluxos de treball més complexos, en què fins i tot cal utilitzar la consola per configurar certs serveis i aplicacions fent que sigui més complicat familiaritzar-se amb la plataforma.

Per altra banda, mentre que les tres plataformes de codi obert analitzades destaquen per les grans possibilitats de personalització, sobretot en l'aspecte d'interfície d'usuari, les plataformes dels grans líders tecnològics destaquen per totes les eines que disposen d'anàlisi i tractament de les dades amb aplicacions avançades d'intel·ligència artificial i aprenentatge automàtic, la qual cosa les fa molt adequades per obtenir informació valuosa i així prendre accions i decisions de negoci confiables.

2.7 Riscos i reptes de les solucions IoT

Encara que les tecnologies IoT s'estan estenent de forma molt ràpida en molts àmbits de la societat, cal tenir en compte que el seu ús també implica uns riscos i reptes que han de ser identificats i gestionats. Alguns d'aquests, aplicats en el sector del transport, són els següents:

- **Alimentació:** els dispositius sense fils necessiten alimentació i com més dades comparteixen, més energia consumeixen. Moltes vegades, segons on estigui instal·lat el dispositiu IoT, no es pot assegurar la disponibilitat d'una font d'alimentació externa o les bateries no es poden canviar fàcilment. Per això, és important que disposin de suficient capacitat d'emmagatzematge o d'un sistema de generació d'energia alternatiu juntament amb una infraestructura adequada per poder supervisar el seu estat.

- **Seguretat:** interceptar la comunicació dels dispositius IoT per recopilar informació sobre béns i clients i interrompre les xarxes per interferir les operacions poden ser objectius temptadors pels ciberdelinqüents. Segons l'estudi [199], el 70% dels dispositius IoT més utilitzats contenen vulnerabilitats relacionades amb la privadesa, autorització insuficient, falta de xifratge en el transport i interfície web insegura. Com que aquests dispositius tenen un paper cada cop més crític en les activitats empresarials, la seguretat des del disseny és important a l'hora de seleccionar qualsevol solució IoT.

- **Disponibilitat de la infraestructura:** les activitats del transport i la logística solen tenir lloc a grans àrees geogràfiques de manera que caldrà tenir en compte que la tecnologia sense fils escollida estigui disponible allà on es necessiti. Moltes de les tecnologies sense fil depenen d'infraestructures de tercers i cap d'elles ofereix una cobertura completa. Així, moltes vegades serà necessari recórrer a una tecnologia sense fils secundària juntament amb un procés que s'adapti al funcionament parcial o complet quan no hi hagi cobertura disponible.

- **Accés a la xarxa i itinerància:** l'ús d'infraestructura sense fils de tercers com les xarxes 5G i NB-IoT han de ser capaços de funcionar a diferents freqüències. Aquestes freqüències no varien només segons la tecnologia, sinó també d'un país a un altre. Així, les empreses han de pagar a cada operador de xarxa per l'accés, normalment a nivell de país i, per a l'accés internacional, a través dels acords d'itinerància, els quals poden suposar un cost addicional substancial.

- **Estàndards i protocols:** amb l'aparició continua de noves solucions, els proveïdors ofereixen productes basats en estàndards establerts i emergents o solucions propietàries. Aquesta proliferació d'ofertes diferents i la manca d'estandardització pot dificultar l'adopció generalitzada d'una tecnologia ja que cap organització vol invertir en una solució que posteriorment sigui difícil d'escalar o de donar suport a llarg termini. Per això, és recomanable que la tecnologia escollida estigui basada en un estàndard publicat amb el suport d'un dels principals grups de la indústria internacional o organismes estàndard com l'IEEE o 3GPP.

- **Normativa reguladora:** tot i que la majoria de les principals tecnologies sense fil es basen en estàndard globals, la normativa reguladora segueix estant molt fragmentada. La disponibilitat de l'espectre de ràdio, per exemple, pot variar segons la regió. Per altra banda, el procés per obtenir l'aprovació pel funcionament de dispositius també és complicat. Els països tenen requisits de prova i certificació molt diferents, cosa que obliga als dispositius sense fil a sotmetre's a un procés d'aprovació llarg i, de vegades, costós per a cada territori d'operació previst.

3. Descripció i arquitectura del sistema

En aquest projecte, es realitza una solució IoT per a la gestió de flotes de vehicles de transport frigorífic per carretera. La definició d'aquesta solució es pot realitzar a partir d'una arquitectura de tres capes on es realitzen diverses operacions.

La primera capa fa referència als dispositius (node físic i nodes simulats) els quals s'han dissenyat específicament per realitzar quatre funcions bàsiques: capturar les dades, processar-les, enviar-les i interactuar.

La segona capa és la de comunicació, la qual possibilita l'intercanvi de dades entre la capa del dispositiu i la capa de servei. Engloba tota la infraestructura que proporciona els mitjans que permeten aquesta comunicació, que en aquest cas, serà bidireccional.

Finalment, la tercera capa és la que fa referència al servei o aplicació en la qual es tracten, processen, visualitzen i analitzen les dades obtingudes tant del node físic com dels nodes simulats. Aquesta capa de servei s'ofereix des d'una infraestructura al núvol.

Tal com s'observa a la Figura 14, el sistema s'ha realitzat principalment sobre la plataforma al núvol d'Amazon Web Services (AWS) mitjançant la qual es podran visualitzar les dades i actuar sobre l'equip de refrigeració instal·lat als vehicles en temps real (BLOC TEMPS REAL) i realitzar un anàlisi forense de les dades emmagatzemades per extreure conclusions i així, poder fer una gestió adequada de la flota de vehicles (BLOC ANÀLISI HISTÒRIC DE DADES).

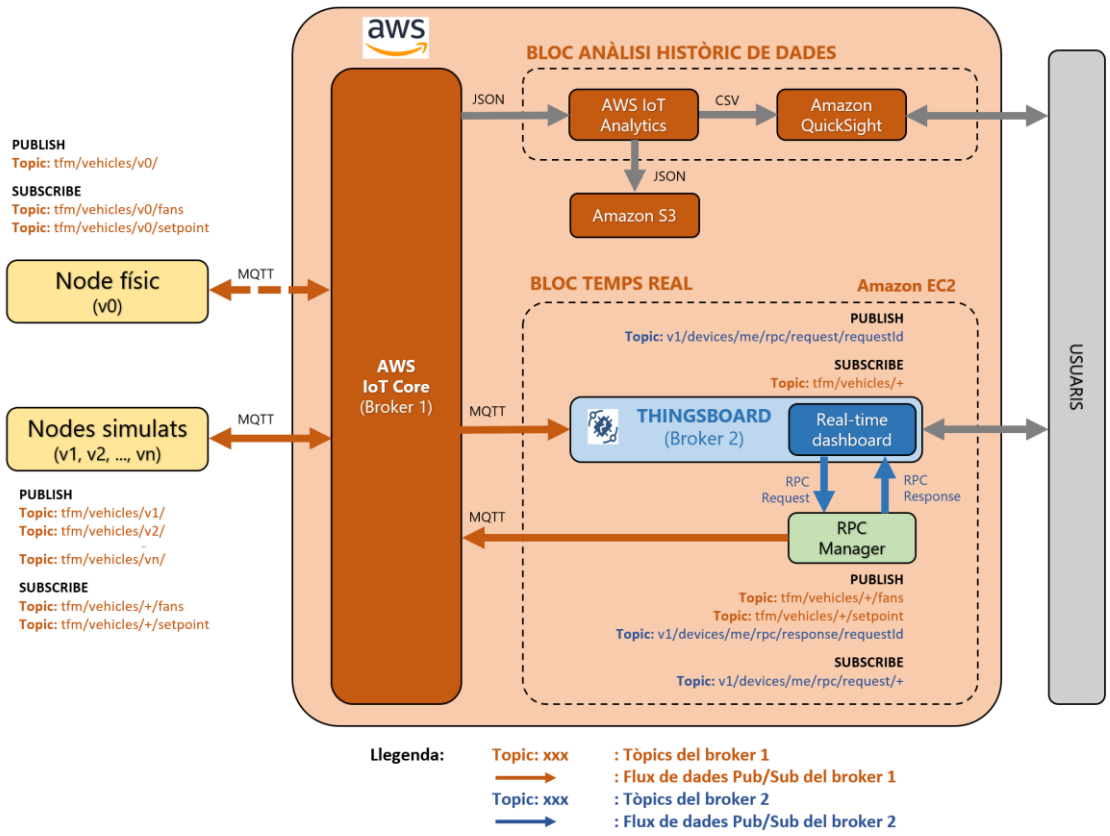


Figura 14. Arquitectura del sistema

Respecte el BLOC TEMPS REAL, s'ha desplegat un servidor virtual Linux mitjançant el servei d'Amazon EC2 que ha permès instal·lar la plataforma IoT de temps real anomenada Thingsboard. Aquesta permet integrar-se amb la plataforma d'AWS, convertir les dades al format requerit i crear de forma ràpida i visual un quadre de comandament amb ginys d'entrada i sortida per tal que l'usuari pugui interactuar amb els nodes. Sobre el mateix servidor virtual, també s'ha desenvolupat un programa (BLOC RPC MANAGER) per tal de poder enviar les ordres cap als nodes, i així variar alguns paràmetres com la temperatura de consigna o l'estat de l'interruptor de l'equip de refrigeració des de la plataforma IoT.

Respecte el bloc d'anàlisi de l'històric de dades, els serveis proporcionats per AWS permeten recopilar, preprocessar i emmagatzemar les dades per tal que els usuaris a través d'una interfície gràfica puguin realitzar un anàlisi forense visualitzant i filtrant les dades rebudes dels dispositius IoT.

A continuació es fa una breu descripció de cadascun dels blocs presents a l'arquitectura:

- **Node físic (v0):** representa el prototip desenvolupat. Captura 8 dades d'interès: la temperatura exterior, la temperatura interior del remolc, la ubicació (latitud i longitud), la velocitat, l'estat de la porta del remolc (oberta/tancada), la temperatura de consigna i l'estat de l'equip de refrigeració (engegat/aturat); i les envia utilitzant el protocol MQTT sobre la tecnologia de xarxa sense fils GPRS.

- **Nodes simulats (v1, v2, ...):** les dades es generen de forma sintètica mitjançant una aplicació desenvolupada en Python. Envia les mateixes dades que el node físic juntament amb l'autonomia i el consum instantani. També s'utilitza el protocol MQTT.

- **AWS IoT Core:** broker MQTT encarregat de filtrar (mitjançant el tòpic) els missatges que són rebuts des dels publicadors (node físic, nodes simulats i RPC Manager) i discriminar així a quins clients subscrits (node físic, nodes simulats i Thingsboard) són entregats. També és l'encarregat d'enviar totes les dades en format JSON cap al bloc d'anàlisi de l'històric de dades.

- **Thingsboard:** té una doble funcionalitat: actuar de broker MQTT per gestionar les peticions creades pels ginys de control dels *dashboards*; i de client subscrit al broker d'AWS IoT Core per poder rebre i representar les dades dels nodes als *dashboards* (*Real-Time dashboard*).

- **RPC Manager:** intermediari entre els dos brokers: Thingsboard i AWS IoT Core. Actua de subscriptor per Thingsboard per rebre els missatges de les peticions i actua de publicador a AWS IoT, per enviar aquests missatges als dispositius.

- **AWS IoT Analytics:** recopila, preprocessa i emmagatzema les dades dels dispositius IoT redirigides des del servei d'AWS IoT Core en format JSON.

- **Amazon S3:** magatzem on es guarden totes les dades recopilades dels dispositius en format JSON.

- **Amazon Quicksight:** permet visualitzar, analitzar i filtrar les dades per realitzar un anàlisi forense a partir dels *datasets* creats a AWS IoT Analytics en format .csv (valors separats per comes).

Per tant, com es pot veure, l'arquitectura de la solució IoT està organitzada al voltant dels dos brokers de la comunicació MQTT: AWS IoT Core i Thingsboard.

Els dispositius publiquen les dades d'interès del vehicle al tòpic corresponent: el node físic les publica al tòpic *tfm/vehicles/v0* i els nodes simulats al tòpic corresponent al vehicle, és a dir, el vehicle 1 al tòpic *tfm/vehicles/v1*, el vehicle 2 al tòpic *tfm/vehicles/v2,* i així successivament amb la resta de vehicles simulats. Com que Thingsboard està subscrit al tòpic *tfm/vehicles/+*, el broker d'AWS IoT Core li reenvia tots els missatges dels nodes.

En el *dashboard* de Thingsboard es representen aquestes dades rebudes dels dispositius, tal com es mostra a la Figura 15, però també es poden enviar missatges cap als dispositius mitjançant el sistema de petició/resposta propi basat en MQTT del que disposa Thingsboard. Cada vegada que l'usuari acciona un giny de control del *dashboard*, com la temperatura de consigna o l'interruptor de l'equip de refrigeració d'un vehicle, Thingsboard publica un missatge amb el nou paràmetre al tòpic *v1/devices/me/rpc/request/requestId*.

Com que el bloc RPC Manager està subscrit al tòpic *v1/devices/me/rpc/request/+*, el broker de Thingsboard li reenvia aquest missatge. Llavors, RPC Manager és l'encarregat d'extreure aquest paràmetre del missatge i publicar-lo al tòpic *tfm/vehicles/+/fans* si s'ha modificat l'estat de l'interruptor de l'equip de refrigeració o al tòpic *tfm/vehicles/+/setpoint* si s'ha modificat la temperatura de consigna. El símbol + és el que identifica el vehicle del qual s'ha modificat aquest paràmetre, de manera que, com que els nodes estan subscrits a aquests tòpics, el broker d'AWS IoT Core li reenvia aquest missatge al vehicle corresponent.

Així, per exemple, si des del dashboard s'ha modificat la temperatura de consigna del vehicle 0, el nou valor establert es publica al tòpic *v1/devices/me/rpc/request/requestId* i és rebut per l'RPC Manager. A continuació, l'RPC Manager identifica des de quin vehicle s'ha fet la petició i publica el valor al tòpic *tfm/vehicles/v0/setpoint*. Aquest valor serà rebut pel node físic ja que és el que està subscrit a aquest mateix tòpic.

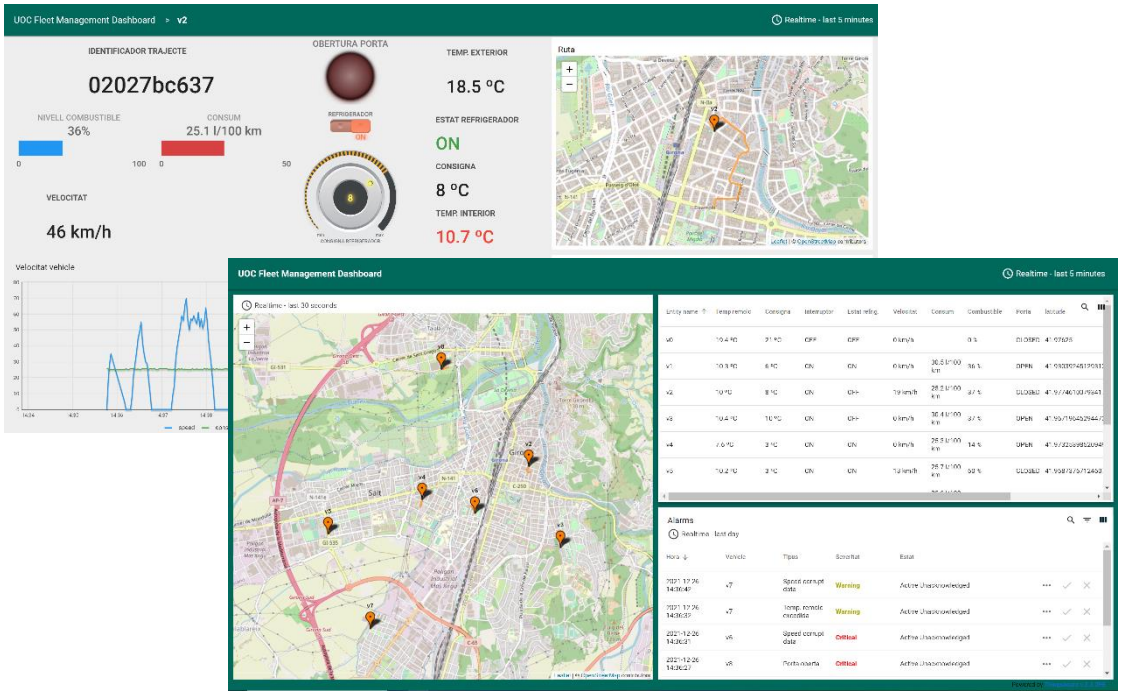


Figura 15. Dashboards creats mitjançant Thingsboard

4. Node físic

En aquest capítol es descriu, a nivell de maquinari i programari, el prototip desenvolupat el qual, per una banda permet obtenir les dades d'interès del vehicle i enviar-les a la plataforma IoT; i per l'altre, actuar sobre l'equip de producció de fred del propi vehicle a partir dels paràmetre rebuts des de la plataforma IoT.

Les dades del vehicle que s'envien a la plataforma són la temperatura exterior, la temperatura interior del remolc, la ubicació del vehicle (latitud i longitud), la velocitat i la detecció de porta oberta del compartiment frigorífic. Les dades rebudes de la plataforma són l'estat de l'interruptor d'engegada i aturada de l'equip de refrigeració i la temperatura de consigna del remolc.

4.1 Maquinari del node físic

En aquest apartat es detalla tot allò referent al maquinari del node físic: els components utilitzats pel seu desenvolupament, l'esquema del circuit i la configuració del mòdul de comunicacions.

4.1.1 Components

Els principals elements que s'han utilitzat per a implementar aquest node físic, mostrat a la Figura 16, són la placa microcontrodora STM32 Nucleo-F446RE, encarregada de recollir, tractar i enviar les dades als perifèrics; i el mòdul de comunicacions SIM7000E, responsable de la comunicació sense fils entre la plataforma IoT i el node físic mitjançant la tecnologia GPRS, juntament amb l'obtenció de les dades de posicionament GNSS.

Com a entrades al sistema, s'han instal·lat dos sensors de temperatura DS18B20 per capturar la temperatura exterior i la temperatura del compartiment frigorífic, i un sensor magnètic MC-38 per detectar l'obertura de la porta d'aquest compartiment. Com a sortides, s'ha instal·lat una cel·la Peltier per simular el sistema refrigerador i s'ha afegit una petita pantalla OLED per visualitzar algunes de les dades d'interès des del propi node físic.

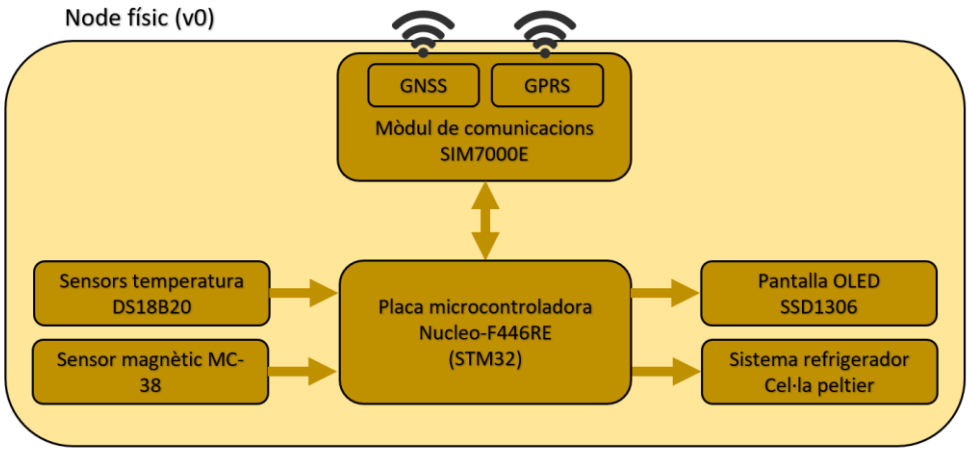


Figura 16. Principals components del node físic
Elaboració pròpia

A continuació, es descriuen en més detall cadascun d'aquests elements citats.

- **Placa STM32 Nucleo-F446RE**

En aquest projecte, com a placa microcontroladora s'ha utilitzat la Nucleo-F446RE, la qual treballa amb el microcontrolador ARM Cortex M4 STM32F446RE, perquè ja es disposava d'aquest model.

Pel desenvolupament d'un primer prototip, com és en aquest cas, es considera una placa molt adequada ja que disposa d'una gran quantitat d'entrades i sortides, una gran varietat d'interfícies de comunicació (USART, I2C, SPI...), bona velocitat de processament (fins a 180 MHz), integra el depurador/programador mitjançant connexió USB i és un dispositiu de baix cost. No obstant, cal destacar que existeixen models inferiors que poden complir perfectament amb els requeriments d'aquest projecte.

Els requeriments mínims necessaris per a que el projecte desenvolupat funcioni correctament amb un microcontrolador STM32 són els següents:

- 1 sortida digital que permeti interrupció externa: utilitzat per detectar el canvi d'estat del sensor magnètic.
- 5 sortides digitals: 2 sortides per connectar els 2 LEDS que serveixen per indicar l'obertura de la porta i quan la unitat de refrigeració està en funcionament; 2 sortides per connectar els dos sensors de temperatura i 1 sortida per connectar la cel·la Peltier.
- 2 USARTS (RX/TX): 1 USART per connectar el mòdul de comunicacions SIM7000E i una altra USART per connectar amb la consola sèrie de l'ordinador (utilitzat exclusivament durant la depuració del codi del programa).
- 1 bus I2C (SCL/SDA) per connectar la pantalla OLED SSD1306.

El port USB del que disposa la placa s'utilitza durant el desenvolupament del projecte com a alimentació de la placa i per descarregar i depurar el codi mitjançant una consola de comunicació sèrie instal·lada a l'ordinador.

Quan a memòria, el codi requereix uns 6 KB de RAM i uns 50 KB de memòria FLASH aproximadament. I quan a velocitat de processament, ajustant el rellotge del microcontrolador a 50 MHz és suficient per a la correcte execució del programa creat.

- **Sensors de temperatura DS18B20**

Per mesurar tant la temperatura exterior del vehicle com la temperatura interior del remolc, s'ha escollit el sensor DS18B20, fabricat per la companyia Maxim Integrated.

És un sensor econòmic que permet un rang ampli de mesura (de -55ºC a 125ºC) i amb una resolució configurable de fins a 0.0625ºC. Es comercialitza tant en forma d'integrat TO-92 com en forma de sonda impermeable, mostrat a la Figura 17, la qual cosa permet realitzar mesures de temperatura en líquids o gasos.

Està format per un processador amb múltiples mòduls, que s'encarreguen de controlar la comunicació, mesurar la temperatura i fins i tot, gestionar un sistema d'alarmes que permet gravar en memòria els límits inferiors i superiors de temperatura.

Figura 17. Sensor de temperatura DS18B20
Extret de naylampmechatronics.com

Aquest sensor destaca principalment per utilitzar el bus de comunicació anomenant 1-Wire, propietat de l'empresa *Maxim Integrated*. El principal avantatge d'aquest bus és que només necessita un únic conductor per realitzar la comunicació (sense comptar el conductor a massa). Els dispositius poden estar alimentats directament per la línia de dades, o mitjançant una línia addicional amb una tensió d'entre 3 i 5.5V. A més, el bus 1-Wire permet connectar tants sensors com es desitgi, tal i com es mostra a la Figura 18, i utilitzar cables més llargs que altres sistemes abans que es deteriori la comunicació.

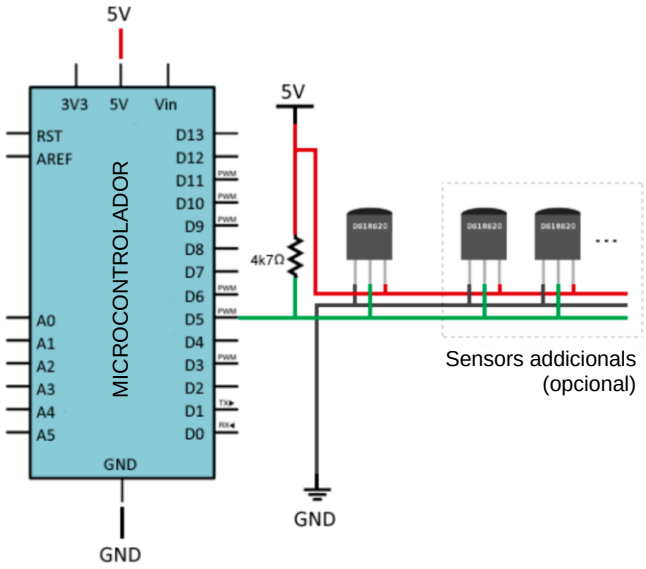


Figura 18. Connexió de sensors al Bus 1-Wire
Adaptat de www.luisllamas.es/temperatura-liquidos-arduino-ds18b20/

D'aquesta manera el sistema es pot escalar fàcilment sense requerir ni haver de configurar noves entrades en el microcontrolador. L'inconvenient, però, és que requereix d'un codi complex, que pot suposar una alta càrrega del processador per consultar l'estat dels sensors. El temps d'adquisició total d'una mesura a una resolució màxima de 12 bits pot ser de fins a 750 ms.

Per disposar de múltiples dispositius en el bus 1-Wire, cada sensor disposa d'una memòria ROM que és gravada de fàbrica amb un número de 64 bits, de manera que els dispositius que es poden connectar en una mateixa xarxa 1-Wire és molt elevada.

En aquest projecte s'han escollit els sensors en forma de sonda impermeable per poder-los ubicar fàcilment a l'interior del remolc del prototip. També s'ha optat per alimentar-los mitjançant una línia addicional de tensió i no per la pròpia línia de dades per facilitar la seva programació i pel fet que aquesta mateixa línia s'utilitza per alimentar altres dispositius.

Per altra banda, pel desenvolupament d'aquest projecte, s'han utilitzat les funcions bàsiques de lectura, escriptura i inicialització del sensor de [21], les quals ja estan adaptades a la família de microcontroladors STM32. A partir d'aquest codi, s'han creat les següent funcions necessàries pel correcte funcionament del prototip del projecte:

- **Funció *ReadROMCode*:** permet obtenir el número identificatiu de 64 bits (codi ROM) de cadascun dels sensor de temperatura per poder-los connectar al mateix bus 1-Wire. Per a que funcioni, només hi pot haver un sensor connectat a la vegada en el microcontrolador.

- **Funció *changeResolutiontemp*:** permet canviar la resolució del sensor. Segons la pàgina 9 del full de característiques del sensor [22], la resolució del sensor és configurable a 9, 10, 11 o 12 bits, permetent així una resolució en temperatura de 0.5ºC, 0.25ºC, 0.125ºC o 0.0625ºC, respectivament. No obstant, a més resolució més llarg és el temps d'adquisició. En el cas d'aquest projecte s'ha considerat suficient una resolució de 10 bits (0.25ºC), aconseguit reduir així el temps d'adquisició a 187.5 ms. Per a que funcioni, només hi pot haver un sensor connectat a la vegada en el microcontrolador.
- **Funció *readTemperature*:** permet obtenir la lectura de temperatura del sensor. Com que s'ha utilitzat el bus 1-Wire, caldrà seguir la seqüència de transaccions especificada a la pàgina 10 del full de característiques per obtenir la temperatura de cadascun dels sensors. Bàsicament caldrà realitzar 3 passos: un procés d'inicialització per indicar que es vol enviar una nova comanda, l'enviament d'una comanda ROM per indicar de quin sensor es vol obtenir la lectura i finalment l'enviament d'una instrucció per indicar que es vol obtenir el valor de temperatura del sensor indicat anteriorment.

Totes aquestes funcions s'han integrat dins la llibreria *sensor_temp.c*.

- **Sensor magnètic MC-38**

Per detectar l'obertura de la porta del remolc s'ha utilitzat un sensor magnètic, l'MC-38, mostrat a la Figura 19. Aquest sensor consta d'un imant i un interruptor magnètic (*reed switch*) en què el primer es fixa a la porta i el segon al marc de la porta.

Figura 19. Sensor magnètic MC-38
Extret de www.amazon.com

Pot funcionar com un interruptor normalment tancat (NT) o com un interruptor normalment obert (NO) mentre les dues parts estan juntes i hi ha camp magnètic. Tanmateix, se solen utilitzar els NT per tal que es generi la condició d'alarma quan algú talla el cable amb la intenció d'inhibir el sistema. És important que no s'utilitzi en portes metàl·liques de material ferromagnètic ja que podria alterar la sensibilitat del sensor.

El seu funcionament és senzill. Centrant-nos amb l'NT, quan s'obre la porta del remolc, se separen les dues parts del sensor i el circuit elèctric s'obre fent desaparèixer el camp magnètic generat per l'imant del sensor. Gràcies a aquest canvi d'estat, és possible detectar l'obertura de la porta.

Pel desenvolupament d'aquest projecte s'ha utilitzat un sensor magnètic MC-38 normalment obert, és a dir, que el circuit es tanca quan l'imant s'allunya (s'obre la porta) i el circuit s'obre quan l'imant s'apropa (es tanca la porta). En aquest cas, la distància de commutació es troba entre els 15 i 25 mm.

La detecció d'aquest canvi d'estat es realitza a partir d'interrupció externa d'un dels pins del microcontrolador STM32, en aquest cas, el pin PB4, de manera que quan es detecta un canvi, s'activa la interrupció i es canvia el valor de la variable *openDoor*, per indicar una obertura (1) o tancament de la porta (0).

- **Mòdul de comunicacions SIM7000E**

El mòdul de comunicacions SIM7000E de *Waveshare*, mostrat a la Figura 20, és l'encarregat tant de rebre les dades de posicionament GNSS com d'enviar la posició i la resta de dades d'interès del vehicle a la plataforma IoT. És compatible amb les tecnologies NB-IoT, Cat-M, EDGE i GPRS, així com amb el posicionament GNSS dels sistemes satel·litals GPS, GLONASS, BeiDou i Galileo.

Figura 20. Mòdul de comunicacions SIM7000E
Extret de www.waveshare.com

Cal tenir en compte que la lletra final del model del mòdul és la que indica la regió per a la qual està destinat: la E significa que és per la regió d'Europa/Àfrica/Australia/Sudest asiàtic, mentre que una C indica que és per a la regió de Xina i una G seria per un abast global. També cal tenir present que el voltatge de la ranura de la targeta SIM sigui de 3V, que és el voltatge que utilitzen actualment les targetes SIM a Europa.

Gràcies a la seva reduïda mida, la baixa latència i l'àmplia cobertura és una opció ideal per a aplicacions IoT com el seguiment de vehicles. A més, com que l'enviament i recepció de dades entre el microcontrolador i aquest mòdul es fa mitjançant comunicació sèrie, el fa compatible amb gran quantitat de dispositius hardware com Raspberry Pi, Arduino, o com és el cas d'aquest projecte, amb STM32, amb velocitats de transmissió de fins a 3686400 bps.

Un altre avantatge d'aquest mòdul és que suporta el protocol de comunicació MQTT, ja que disposa d'instruccions específiques que faciliten l'enviament i recepció de dades mitjançant el model publicació/subscripció en el qual està basat MQTT.

Finalment, també cal destacar que aquest mòdul és compatible amb les comandes de control anomenades comandes AT (de la paraula *Attention*). Es considera un estàndard de comunicació adoptat en telefonia mòbil que permet realitzar una sèrie d'accions com consultar l'estat de connexió de la xarxa, introduir el PIN de la targeta SIM, definir el tipus de connexió, establir el servidor APN, entre d'altres.

Aquestes comandes són les que s'envien des del microcontrolador a aquest mòdul a través del port sèrie, per tal de poder, per exemple, establir connexió a la xarxa GPRS, capturar la posició GNSS i enviar les dades d'interès del vehicle a la plataforma IoT. En capítols posteriors es detallen quines comandes s'han utilitzat per a la realització d'aquest projecte i com s'ha implementat la comunicació entre el microcontrolador i aquest mòdul de comunicacions.

Algunes de les especificacions més destacades que s'han tingut en compte pel desenvolupament del projecte es presenten a la Taula 6.

Característics comunicació mòbil	
Banda de freqüències	**NB-IoT/Cat-M:** FDD-LTE B3/B8/B20/B28 **GPRS/EDGE:** 900/1800 MHz
Velocitat de dades	**NB-IoT:** 34 (DL) / 66 (UL) **Cat-M:** 300 (DL) / 375 (UL) **EDGE:** 236.8 (DL) / 236.8 (UL) **GPRS:** 85.6 (DL) / 85.6 (UL)
Targeta SIM suportada	**NB-IoT:** targeta SIM específica NB 2FF Estàndard **Cat-M/GPRS/EDGE:** targeta SIM normal 2FF Estàndard
Característiques GNSS	
Sistemes satel·litals	GPS, BeiDou, GLONASS, Galileo
Tipus de recepció	16 canals, codi C/A
Time to First Fix (TTF)	En fred: < 35 segons; en calent: < 1 segon
Precisió horitzontal	< 2.5 m
Altres	
Voltatge d'alimentació	5 V
Comandes de control	Comandes AT (3GPP TS 27.007, 27.005 i SIMCOM millorat)
Protocols suportats	TCP, UDP, PPP, HTTP, FTP, MQTT, CoAP, TLS, DTLS...
Connectors d'antena	Antena LTE (SMA) + Antena GNSS (IPEX)
USB	Testatge de comandes AT, actualització de firmware...
Ranura targeta SIM	1.8V / 3V

Taula 6. Principals especificacions del mòdul de comunicacions SIM7000E

Cal tenir en compte que aquest mòdul de comunicacions no es pot alimentar directament mitjançant una de les sortides de 5V de les quals disposa el microcontrolador ja que no proporcionen suficient intensitat pel correcte funcionament del mòdul. Per tant, és necessària una font d'alimentació externa de 5V que subministri un mínim de 1.5A.

Per veure la resta d'especificacions, la distribució i definició dels pins, la ubicació dels diferents elements sobre la placa del mòdul o el diagrama del circuit, entre d'altres, es pot accedir a la pàgina web del seu distribuïdor a través de [23].

- **Sistema de refrigeració**

El sistema refrigerador és l'encarregat de fer baixar la temperatura del remolc del vehicle per conservar els elements transportats a la temperatura desitjada. Les unitats de refrigeració dels vehicles professionals solen ser equips de fred formats per un compressor, una unitat de condensació i un evaporador. És evident que per a la realització d'aquest projecte no es disposa d'un equip d'aquestes característiques.

Així doncs, com que l'objectiu és en realitat aconseguir controlar la temperatura d'un compartiment frigorífic a partir d'una consigna enviada des de la plataforma IoT, s'ha considerat oportú utilitzar una cel·la Peltier com a sistema equivalent i poder així demostrar el correcte funcionament dins la solució IoT.

Una cel·la Peltier, l'aspecte de la qual es mostra a la Figura 21.a, té la propietat de que quan passa intensitat a través d'ella, un costat de l'element es refreda i l'altre s'escalfa. Requereixen d'una gran quantitat de corrent (generalment entre 2 i 6 A a 12 V) però tenen l'avantatge respecte els refrigeradors convencionals que no tenen parts mòbils que puguin fallar. Es poden trobar en petites neveres de càmping i refrigeradors de begudes.

El principal problema de les cel·les Peltier és que el costat fred i calent estan molt junts. És per això que normalment es col·loquen dissipadors de calor i ventiladors en ambdós costats de la cel·la Peltier per a que funcioni de manera més eficient, com es pot veure a la Figura 21.b.

a) b)

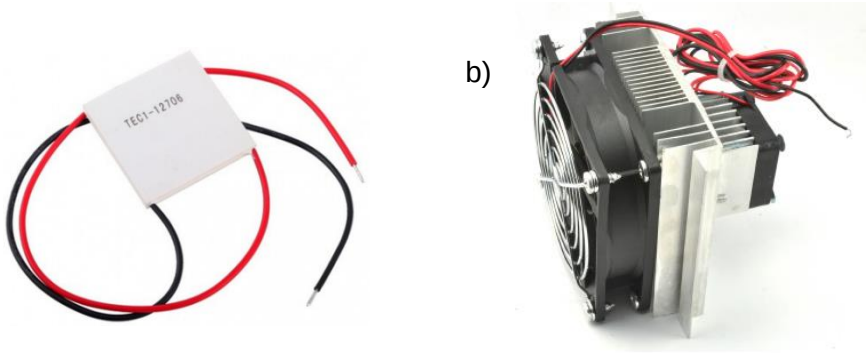


Figura 21. a) Cel·la Peltier. b) Unitat de refrigeració Peltier
Extret de www.amazon.com

Aquesta unitat de refrigeració Peltier amb doble dissipador i ventilador és la que s'ha utilitzat en aquest projecte. Quan la temperatura de l'interior del remolc sigui superior a la consigna donada des de la plataforma IoT, s'enviarà una senyal des del microcontrolador per activar la unitat de refrigeració. Quan la temperatura de l'interior del remolc sigui inferior a la consigna, s'enviarà l'ordre per a que s'apagui.

No obstant, per tal que la unitat de refrigeració no s'estigui encenent i apagant contínuament cada vegada que la temperatura de l'interior del remolc difereix de la consigna, es realitzarà un control per histèresi, la qual cosa equival a tenir dos llindars de temperatura en lloc d'un, tal i com es pot veure a la Figura 22. Així, si la temperatura de l'interior del remolc sobrepassa el llindar superior, la unitat de refrigeració s'encén i només s'apagarà quan sobrepassi el llindar inferior. D'aquesta manera, la inèrcia natural del sistema s'utilitza per introduir retards en la commutació.

Evidentment, els valors de consigna i d'histèresi vindran determinats pel tipus d'elements transportats en el vehicle i del marge de temperatures que es pugui assumir durant el seu transport. En el cas d'aquest projecte, s'ha considerat oportú disposar d'una temperatura de consigna seleccionable des de la plataforma IoT per poder demostrar la correcta comunicació bidireccional entre els nodes i la plataforma mitjançant el protocol MQTT, però la histèresi s'ha fixat a ±1ºC.

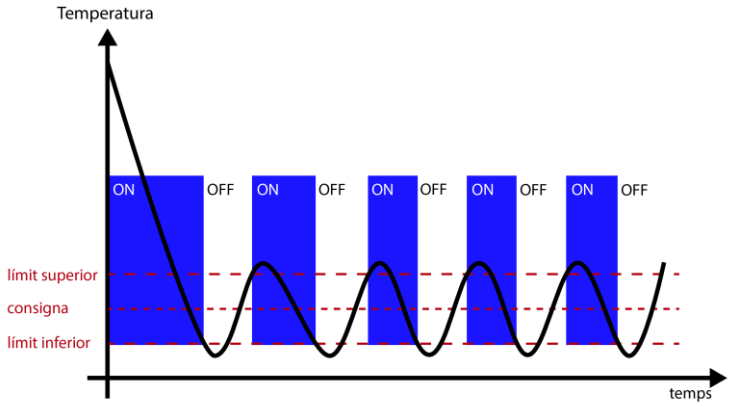


Figura 22. Histèresi en una unitat de refrigeració
Elaboració pròpia

Per altra banda, també cal tenir en compte que la connexió entre el microcontrolador i la unitat de refrigeració no es pot efectuar directament, tal com es mostra a la Figura 23. Mentre que el microcontrolador treballa a 3.3V/5V i les sortides com a màxim només poden subministrar 25 mA, la unitat de refrigeració requereix 4 A a 12 V. És per això que s'ha utilitzat un MOSFET d'alta potència com a element intermedi juntament amb una font d'alimentació de 12 V amb una intensitat mínima de 5A per permetre activar/desactivar la unitat de refrigeració, formada per la cel·la Peltier i els dos ventiladors.

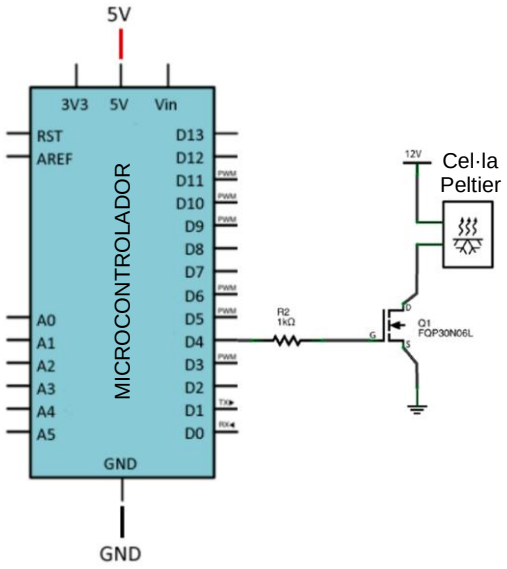


Figura 23. Connexió d'una cel·la Peltier al microcontrolador
Adaptat de www.luisllamas.es/temperatura-liquidos-arduino-ds18b20/

S'ha escollit el MOSFET FQP30N06L, el qual permet suportar fins a 32 A i necessita una baixa tensió de porta per activar-se (a partir de 2 V). La L del final significa que és un MOSFET de nivell lògic la tensió llindar del qual és l'adequada per l'ús amb sortides digitals de 3,3 V, com és el cas del microcontrolador utilitzat en aquest projecte. La resistència entre la porta del MOSFET i del microcontrolador és necessària per assegurar que les corrents de pic que es produeixen quan el MOSFET canvia d'estat no sobrecarreguin el pin GPIO.

A més, aquest MOSFET pot suportar els 4 A requerits per a la unitat de refrigeració sense escalfar-se el suficient com per a necessitar un dissipador de calor.

- **Pantalla OLED**

En el prototip també s'ha incorporat una pantalla OLED de 0.96 polzades com la mostrada a la Figura 24, per poder visualitzar si el procés d'inicialització del sistema (comprovació de la connexió amb el mòdul de comunicacions, cerca de satèl·lits, comprovació de la connexió amb l'APN i amb el servidor d'AWS...) es realitza correctament o si s'ha produït algun problema. Després d'aquest procés, es mostren en temps real algunes dades d'interès del vehicle com la temperatura exterior, la temperatura del remolc, la temperatura de consigna, l'estat del refrigerador (ON/OFF) i l'estat de la porta del remolc (OBERTA/TANCADA).

Figura 24. Pantalla OLED de 0.96 polzades
Elaboració pròpia

Aquestes pantalles OLED ja venen acoblades amb el controlador i tota l'electrònica per a que funcionin correctament. Solen utilitzar un controlador bastant conegut anomenat SSD1306, del qual es poden trobar fàcilment llibreries tant per Arduino, Raspberry Pi i STM32. La comunicació entre el microcontrolador i el SSD1306 es realitza mitjançant I2C en què l'únic que cal conèixer al respecte és la direcció I2C del dispositiu, proporcionada normalment pel propi fabricant. En el cas d'aquesta pantalla del distribuïdor *AZDelivery*, la direcció I2C és la 0x78 la qual es defineix a l'arxiu *ssd1306.h*

Val a dir que, aquesta pantalla també ha estat de gran utilitat per depurar el codi del programa i veure el comportament d'algunes variables durant l'execució del codi.

4.1.2 Esquema del circuit

A la Figura 25 es presenta l'esquema del circuit desenvolupat per a la implementació del node físic.

Id.	Element	Descripció
P1	SIM7000E	Mòdul de comunicacions
P2	SSD1306	Pantalla OLED
P3	Cel·la peltier	-
P4	MC-38	Sensor magnètic
P5, P6	DS18B20	Sensor temperatura

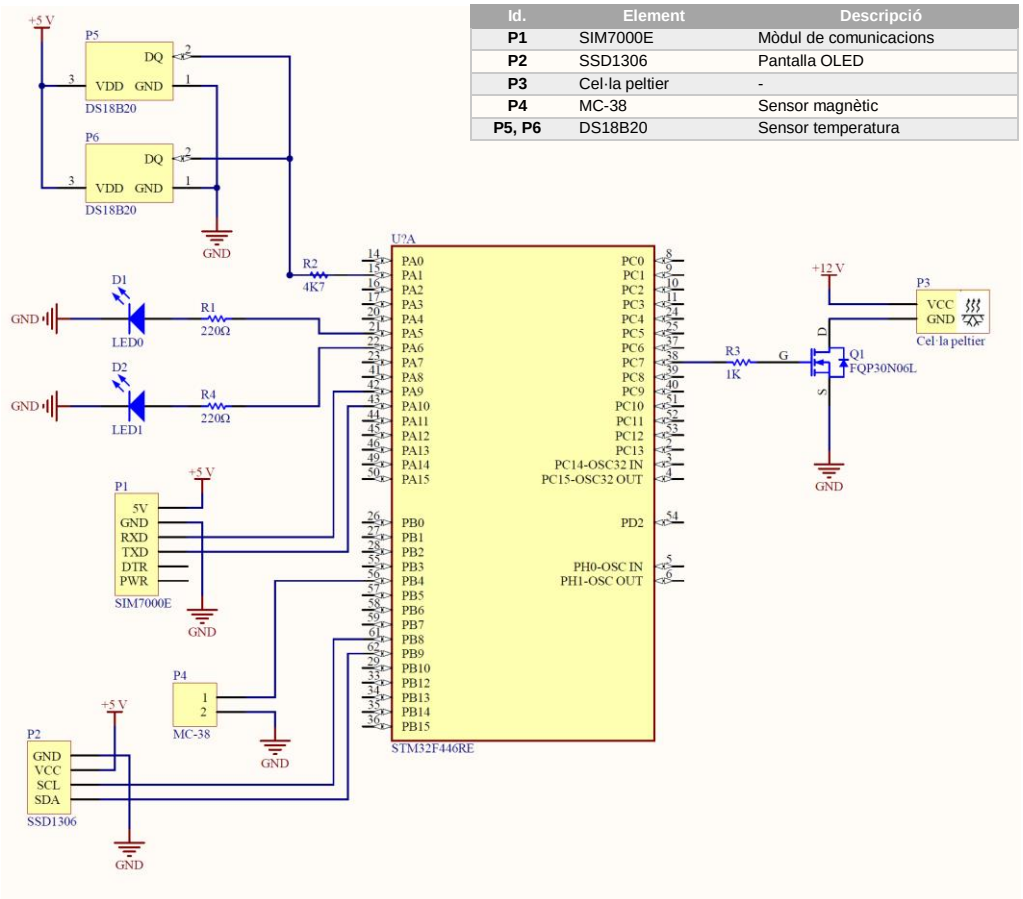


Figura 25. Esquema del circuit del node físic
Elaboració pròpia

Pel seu correcte funcionament es necessita una font d'alimentació de 12V DC que subministri un mínim de 5A per alimentar la cel·la Peltier i els ventiladors i una font d'alimentació de 5V DC que proporcioni un mínim de 2A per alimentar la placa microcontroladora STM32, el mòdul de comunicacions SIM7000E, els sensors de temperatura i la pantalla OLED.

Cal tenir en compte que en aquest esquemàtic no s'ha representat l'alimentació del microcontrolador. El motiu és que la placa microcontroladora de desenvolupament NUCLEO adquirida disposa d'un pin d'alimentació de 5V que permet subministrar directament aquest voltatge. Internament, disposa d'un regulador que baixa la tensió a 3,3V, voltatge al qual treballa realment el microcontrolador.

Els pins utilitzats en aquest projecte es descriuen a la Taula 7:

Pin STM32	Tipus	Funció
PA1	E/S Analògica	Connexió amb els sensors de temperatura DS18B20
PA2*[1]	USART2_TX	Comunicació sèrie amb l'ordinador (exclusivament per depurar el codi del programa desenvolupat) *Baudrate*: 115200
PA3*[1]	USART2_RX	
PA5	Sortida digital	LED indicador estat porta remolc (1 - oberta / 0 - tancada)
PA6	Sortida digital	LED indicador estat equip refrigeració (1 - ON / 0 - OFF)
PA9	USART1_TX	Comunicació sèrie amb SIM7000E *Baudrate*: 115200
PA10	USART1_RX	
PC7	Sortida digital	Connexió amb la cel·la Peltier
PB4	Sortida digital	Connexió amb el sensor magnètic MC-38
PB8	I2C1_SCL	Connexió I2C amb la pantalla OLED SSD1306
PB9	I2C1_SDA	
5 V*[2]	Alimentació	Alimentació de la placa microntroladora a 5V.

Taula 7. Pins del microcontrolador utilitzats

*[1] Encara que en els pins PA2 i PA3 no hi ha res connectat tal i com es pot veure a l'esquema del circuit, són utilitzats internament per la placa de desenvolupament STM32 per a la comunicació sèrie a través de l'USB, corresponent a la USART2.

*[2] Aquest pin no apareix a l'esquema del circuit, no obstant, la placa NUCLEO-F446RE adquirida sí que disposa d'ell.

4.1.3 Configuració del mòdul SIM7000E

El mòdul de comunicacions SIM7000E requereix una configuració prèvia per a que pugui funcionar correctament. Cal seguir els tres passos que es descriuen a continuació per tal de tenir operativa la comunicació entre el mòdul i el servei d'AWS IoT Core:

- **Introduir la targeta SIM**

Cal introduir una targeta SIM (tipus 2FF Estàndard) operativa i amb el servei de dades mòbils activat al sòcol corresponent del mòdul de comunicacions. Cal tenir en compte que en cas d'utilitzar la xarxa NB-IoT, no serveix la targeta SIM utilitzada en els telèfons mòbils sinó que és una targeta SIM específica per a aquesta tecnologia.

Com que el mòdul de comunicacions adquirit també suporta la banda de freqüències GPRS/EDGE, per a les proves inicials s'ha utilitzat una targeta SIM de Telefónica que es pot trobar a qualsevol telèfon mòbil. Per a les proves finals, ja s'ha utilitzat una SIM de Telefónica compatible amb NB-IoT, LTE-M i també GPRS/EDGE, adquirida a través del distribuïdor [24]. Per tal que el sistema es connecti a una xarxa o a una altra, simplement cal canviar uns paràmetres que es detallen en el proper apartat.

- **Verificar comunicació**

Després de connectar les antenes de LTE i GNSS, cal comprovar que el mòdul de comunicacions respon correctament a les comandes AT enviades i es registra correctament a la xarxa corresponent (GPRS, NB-IoT...), assegurant-nos així que la SIM està operativa i el mòdul funciona correctament.

Per això, és recomanable descarregar-se una de les múltiples eines disponibles a la xarxa que permeten comunicar-se amb dispositius d'aquest tipus mitjançant comandes AT. Pel desenvolupament d'aquest projecte s'ha utilitzat el programa *AT Command Tester*, mostrat a la Figura 26, desenvolupat per oakkar7 i disponible a GitHub a través de [25]. És compatible amb el sistema operatiu Windows i ha funcionat correctament amb totes les comandes AT utilitzades en aquest projecte.

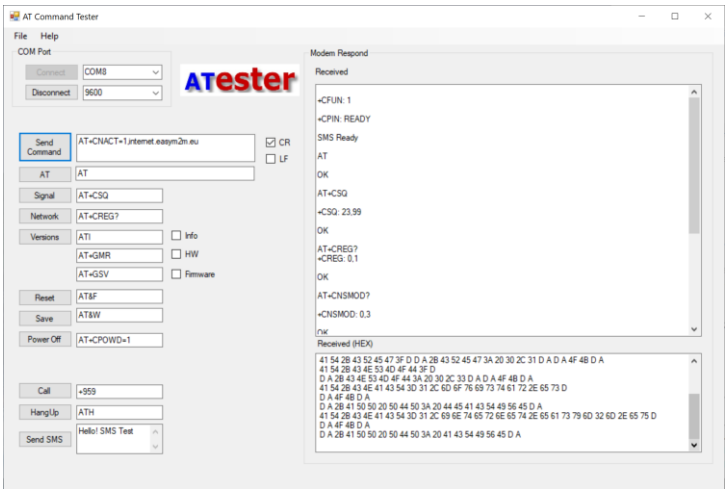


Figura 26. AT Command Tester
Elaboració pròpia

- **Copiar els certificats i claus**

A banda de connectar-se a una xarxa de comunicacions, el dispositiu ha de connectar-se i comunicar-se amb el servei d'AWS IoT Core.

Tal i com s'havia explicat a l'apartat 2.4 Protocols de comunicació en IoT de l'estat de l'art, en el protocol MQTT es pot verificar la identitat de tres maneres diferents: per ID de client, per nom d'usuari i contrasenya i per xifratge TLS. Tenint en compte això, cal saber que el broker d'AWS IoT només és compatible amb aquesta última opció, el xifratge TLS.

Això fa que sigui necessari crear certificats i claus per a aquells dispositius que es vulguin connectar a AWS IoT. Per tant, pel prototip creat, es requereix copiar tres arxius dins la memòria interna del mòdul de comunicacions SIM7000E, procés explicat en detall a l'annex *A.1 Configuració d'AWS IoT Core* d'aquesta memòria.

Un cop realitzats aquests tres passos, ja es podrà connectar aquest mòdul al microcontrolador segons l'esquema del circuit proporcionat anteriorment.

4.1.4 El prototip

Per realitzar les proves en un entorn real, s'ha connectat una font d'alimentació d'ordinador a l'endoll de 220V AC d'un turisme. Aquesta font d'alimentació ja proporciona els voltatges de 12V i 5V adequats per alimentar el prototip creat.

Per a una instal·lació definitiva, els 12V DC s'haurien d'extreure de la bateria del cotxe i seria necessari adquirir un convertidor DC/DC de 12V a 5V, el qual es pot trobar fàcilment per un cost inferior als 10€.

A la Figura 27, es mostra una foto del prototip on es poden identificar els diversos elements comentats en els apartats anteriors.

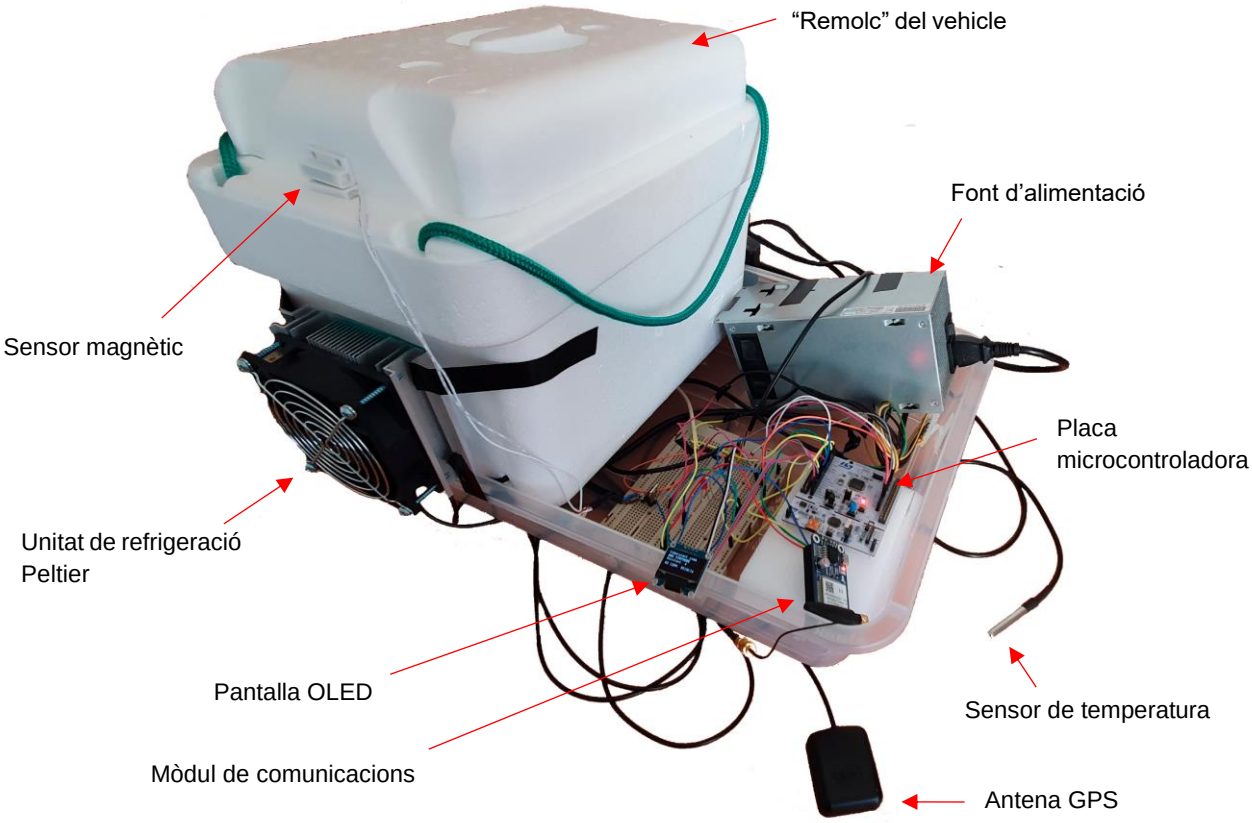


Figura 27. Prototip del node físic
Elaboració pròpia

4.2 Programari del node físic

En aquest aparat es descriu tot allò referent al programari del node físic: des de l'entorn de desenvolupament utilitzat per a la seva programació fins a la descripció a nivell de codi del funcionament del prototip i cadascun dels elements que el formen.

4.2.1 Entorn de desenvolupament (IDE)

Pel desenvolupament del codi de programa del node físic, s'ha utilitzat l'eina de desenvolupament integral que disposa *STMicroelectronics* pels dispositius STM32 anomenada *STM32CubeIDE*, disponible a [26]. Consisteix en una plataforma de desenvolupament en C/C++ que permet la configuració de perifèrics, generació i compilació de codi i funcions de depuració per als microcontroladors i microprocessadors de la família STM32.

Un cop seleccionat el microcontrolador o microprocessador amb el qual es vol treballar, el programa permet crear d'una manera gràfica la plantilla inicial de codi, seleccionant i configurant sobre la pròpia imatge del xip, les entrades i sortides que es volen utilitzar, els perifèrics i la freqüència del rellotge, tal com es mostra a la Figura 28. En qualsevol moment durant el desenvolupament, es poden modificar o configurar noves entrades i sortides i regenerar el codi d'inicialització sense impacte en el codi d'usuari.

Connectant la placa microcontroladora per USB a un ordinador, aquesta és detectada automàticament per Windows i mitjançant el programa *STM32CubeIDE* es pot descarregar i provar directament el codi del programa a la pròpia placa.

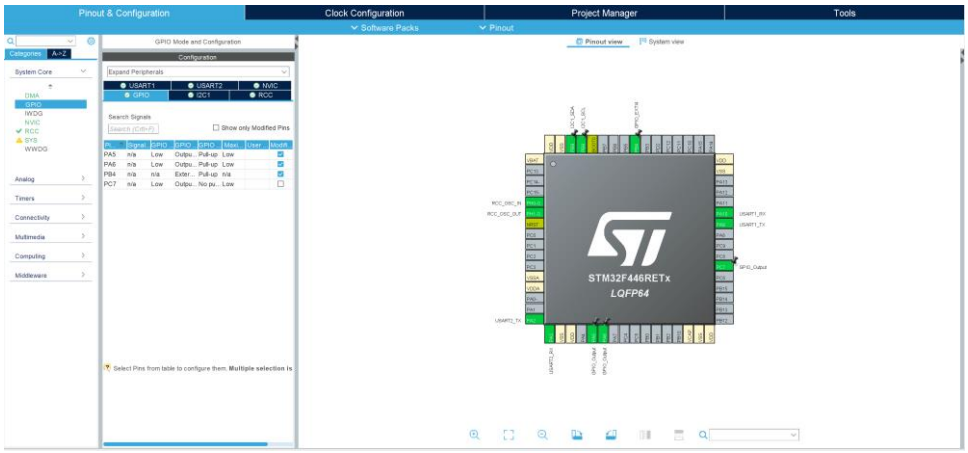


Figura 28. Eina de configuració del dispositiu
Elaboració pròpia

Els arxius i les llibreries creades o utilitzades pel correcte funcionament del prototip es mostren a la Taula 8.

Descripció	Axius .h	Arxius .c
Llibreria sensors de temperatura	DS18B20.h	DS18B20.c
Llibreria pantalla OLED	ssd1306.h fonts.h	ssd1306.c fonts.c
Programa principal	main.h	main.c

Taula 8. Arxius i llibreries necessàries

Pel que fa al mòdul de comunicacions SIM7000E no s'ha trobat cap llibreria compatible amb la placa microcontroladora STM32, és per això que, s'ha desenvolupat tot el codi necessari per a establir la comunicació entre ambdós i pel correcte enviament i recepció de comandes AT.

4.2.2 Funcionament del prototip

El codi consta bàsicament de dues parts ben diferenciades: una primera part de configuració i una segona part d'enviament i actualització de dades de forma periòdica.

En la primera part, concretament, s'inicialitzen els perifèrics del microcontrolador, s'habilita el mòdul de comunicacions i la senyal GPS, es configuren els paràmetres del mòdul per a la connexió a la xarxa de comunicacions corresponent, es configura el protocol MQTT per a que el dispositiu es pugui connectar a AWS IoT i es defineixen els tòpics als quals es subscriu el dispositiu.

A la segona part, un cop la ubicació està fixada, s'entra en el bucle principal del codi on de forma periòdica i seqüencial es realitza la lectura de les temperatures, s'inicia o s'atura l'equip refrigerador segons els paràmetres rebuts de la plataforma IoT, s'actualitza l'estat de la porta del remolc, s'obtenen les dades del GPS, s'actualitza la pantalla OLED amb els nous valors i es publiquen les dades d'interès del vehicle al tòpic MQTT corresponent.

El diagrama de funcionament del prototip es mostra a la Figura 29.

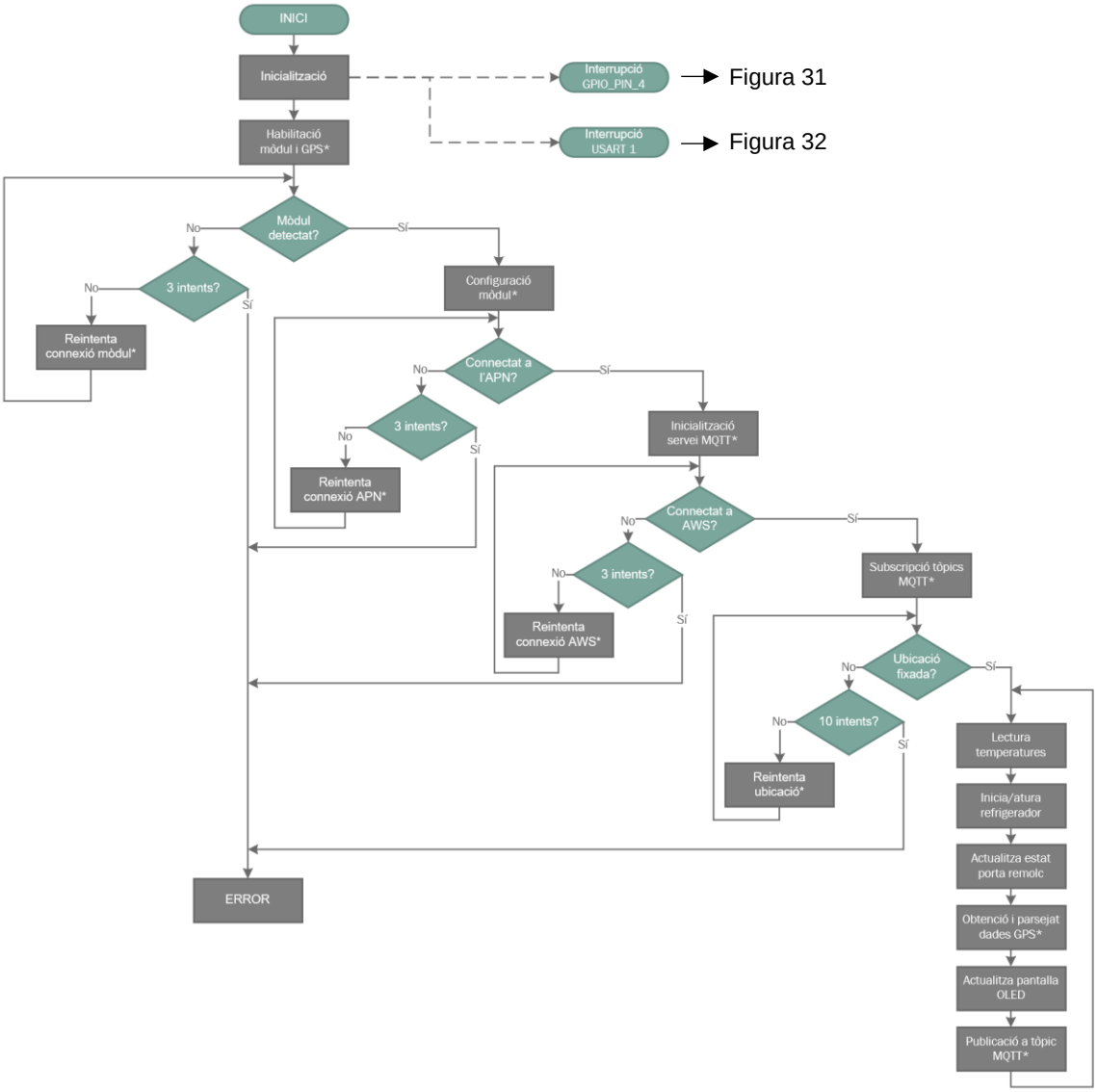


Figura 29. Diagrama de flux del codi de programa del microcontrolador STM32
Elaboració pròpia

A continuació, es descriuen en més detall els aspectes més importants dels diferents blocs del diagrama de flux anterior.

En el bloc d'inicialització, com el seu nom indica, s'inicialitzen tots els perifèrics utilitzats: les USARTS per a la comunicació sèrie amb el mòdul de comunicacions i l'ordinador, ambdós a 115200 bps, la interfície I2C per la pantalla OLED, la pròpia pantalla OLED, els temporitzadors i les interrupcions les quals permeten rebre les dades a través de la USART i detectar l'obertura de la porta amb l'activació del sensor magnètic.

A continuació es comprova que el mòdul de comunicacions respon correctament a les peticions enviades pel microcontrolador mitjançant l'enviament de la comanda *AT*. Si després de tres intents segueix sense respondre, es mostra un error a la pantalla OLED, indicant el problema.

Posteriorment, com que el GPS tarda un temps a fixar la posició (en fred, uns 30 segons; i en calent, 1 segon, aproximadament) s'habilita també en aquest punt la senyal GPS per a que vagi fixant la ubicació mitjançant la comanda *AT+CGNSPWR=1* mentre es realitzen les tasques següents.

El següent pas és enviar al mòdul de comunicacions els paràmetres que han de permetre establir la connexió a la xarxa de comunicacions. Es comprova l'estat de registre a la xarxa (*AT+CREG?*), la qualitat de la senyal (*AT+CSQ*), s'escull el tipus de tecnologia per connectar-se a la xarxa de comunicacions (*AT+CNMP*) i s'activa la connectivitat IP a través de l'APN de l'operador (*AT+CNACT*). Si després de tres intents no s'obté connectivitat, es mostra un error a la pantalla OLED, indicant el problema.

En el següent bloc, es configura tot allò refent al protocol MQTT: es defineix la versió del protocol TLS a utilitzar (*AT+CSSLCFG*), els noms dels certificats i claus copiats dins el mòdul necessaris per a la comunicació xifrada (*AT+SMSSL*), la URL i el port del servidor MQTT (l'*endpoint d'AWS IoT*) i s'estableix la connexió (*AT+SMCONN*). De la mateixa manera que abans, si no s'aconsegueix connectar amb el servidor d'AWS després d'intentar-ho tres vegades, es mostra un error a la pantalla OLED, indicant el problema.

A continuació, el dispositiu es subscriu a dos tòpics (*AT+SMSUB*), corresponents a la consigna de temperatura i a l'estat de l'interruptor de l'equip refrigerador. El procés per capturar els missatges i extreure els valors rebuts està explicat en pàgines posteriors.

Arribats a aquest punt, previ a l'inici de l'enviament continu de dades cap al servei d'AWS IoT, es comprova si el GPS ja té la posició fixada. Per determinar-ho, s'envia la comanda *AT+CGNSINF* i es consulta quin valor apareix a la posició 25 de la trama NMEA rebuda, corresponent al segon paràmetre de la trama anomenat estat de localització (fixada o no fixada). Així, mentre sigui 0 (posició no fixada), es segueix enviant la comanda d'obtenció de la trama NMEA cada quatre segons en bucle fins que valgui 1 (posició fixada). Si després de deu intents, encara no s'ha fixat la ubicació, es mostrarà un error a la pantalla OLED, indicant el problema.

Un cop la ubicació ha estat fixada, s'entra dins un bucle infinit on s'obtindran les dades d'interès del vehicle de la següent manera:

- **Temperatura interior i exterior:** s'executa la funció *readTemperature()* per a cadascun dels sensors per llegir el valor de temperatura.

Gràcies a la llibreria desenvolupada en aquest projecte pels sensors DS18B20 compatible amb els microcontroladors STM32, només cal executar les línies de codi següents per obtenir el valor:

```
Temp_ext = readTemperature (ROM_ID1);
Temp_int = readTemperature (ROM_ID2);
```

En aquest cas, ROM_ID1 correspon al codi ROM del sensor de temperatura exterior i ROM_ID2, correspon al codi ROM del sensor de temperatura situat a l'interior del remolc.

- **Posició i velocitat:** s'envia la comanda *AT+CGNSINF* i de la resposta obtinguda s'extreu la latitud, la longitud i la velocitat.

Per extreure aquests paràmetres, cal analitzar i parsejar la trama GPS rebuda. Un exemple de trama que s'obté cada vegada que s'executa la comanda AT corresponent, es mostra a la Taula 9.

Trama	
+CGNSINF: 1,1,20190408193057.000,41.976182,2.802547,72.800,0.09,219.8,1,,2.2,2.4,1.0,,6,5,,,50,,	
Part trama	**Significat**
1	Estat d'execució (habilitat o no habilitat)
1	Estat de localització (fixada o no fixada)
20190408193057.000	Data i hora en UTC (yyyyMMddhhmmss.sss)
41.976182	**Latitud (±dd.dddddd)**
2.802547	**Longitud (±ddd.dddddd)**
72.800	Altitud sobre el nivell del mar (m)
0.09	**Velocitat (km/h)**
219.8	Rumb (º)
1	Tipus de localització (2D, 3D, no fixat)
-	Reservat
2.2	Dilució de la precisió horitzontal (HDOP)
2.4	Dilució de la posició (PDOP)
1.0	Dilució de la precisió vertical (VDOP)
-	Reservat
6	Nombre de satèl·lits a la vista
5	Nombre de satèl·lits utilitzats
-	Reservat
-	Reservat
50	High Power Amplifier (HPA)
-	Reservat

Taula 9. Paràmetres de la trama GPS

Com es pot veure, els paràmetres estan separats per comes, de manera que cal recórrer tota la trama i anar dividint la cadena de caràcters i guardant el valors d'interès dins de variables.

- **Temperatura de consigna i estat de l'interruptor de l'equip de refrigeració:** definides com a variables globals, s'extreuen del missatge de subscripció rebut per interrupció de la USART1, i s'utilitzen per determinar si s'ha d'engegar o aturar l'equip de refrigeració.

Quan l'usuari modifica un d'aquests dos paràmetres des de la plataforma IoT, o bé, obre el quadre de comandament del vehicle, aquest valor es publica al tòpic corresponent i el broker el reenvia als subscriptors, en aquest cas, al node físic, obtenint un missatge amb el format següent:

+SMSUB: "tfm/vehicles/v0/fans","1"
+SMSUB: "tfm/vehicles/v0/setpoint","17"

En aquest cas, vol dir que l'usuari des de la plataforma IoT ha posat l'interruptor de l'equip refrigerador a ON (1) i ha seleccionat una temperatura de consigna de 17ºC.

Per tant, quan a la USART1 del dispositiu arriba un missatge d'aquest tipus, s'extreu el valor i s'actua en conseqüència, engegant o aturant l'equip de refrigeració segons la nova temperatura de consigna o l'estat de l'interruptor.

El diagrama de blocs que permet determinar si engegar o aturar l'equip de refrigeració es mostra a la Figura 30. Per exemple, si l'interruptor està activat (fan = 1) i la temperatura de consigna s'ha fixat a 10ºC, fins que la temperatura del remolc no estigui per sobre dels 11ºC, no s'activarà l'equip de refrigeració i no s'aturarà fins que la temperatura del remolc hagi baixat per sota dels 9ºC. Per altra banda, si es desactiva l'interruptor (fan = 0), s'aturarà l'equip de refrigeració, independentment de la temperatura del remolc i la consigna.

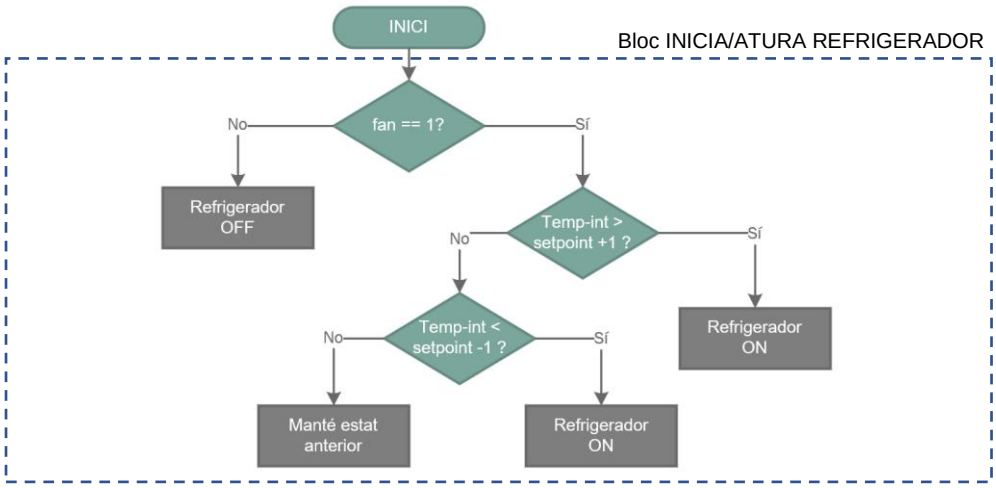


Figura 30. Diagrama de flux de funcionament de l'equip de refrigeració
Elaboració pròpia

- **Estat de l'equip de refrigeració:** el seu valor s'obté en funció de les variables de temperatura de consigna i estat de l'interruptor de l'equip de refrigeració de manera que permet saber si l'equip de refrigeració està en funcionament (Refrigerador ON) o està aturat (Refrigerador OFF).
- **Estat de la porta del remolc:** definida com a variable global, el valor s'extreu de la funció d'interrupció externa tal i com es mostra a la Figura 31. Quan es detecta un canvi d'estat en el PIN4, es comprova si el pin està en estat alt (1) o baix (0). Si està en estat alt significa que s'ha obert la porta i si està en estat baix, que s'ha tancat.

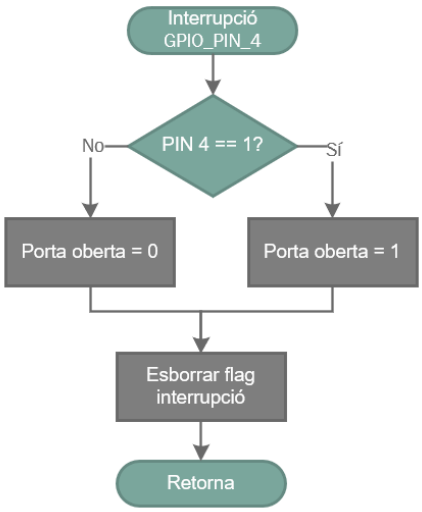


Figura 31. Diagrama de flux de la interrupció externa
Elaboració pròpia

Un cop obtinguts aquests valors, s'actualitza la pantalla OLED amb els nous valors i es publiquen al tòpic *tfm/vehicles/v0* del broker d'AWS IoT en format JSON.

La freqüència en la que es repeteix aquest bucle és la que determina la freqüència en què s'actualitzen les dades per ser enviades a la plataforma IoT. En aquest cas, per provar el correcte funcionament del prototip, s'envien les dades cada deu segons. No obstant, aquest valor pot ser modificat en funció de les necessitats i condicions de seguiment de la flota de vehicles.

Com s'ha pogut veure a les línies anteriors, la comunicació entre el microcontrolador i el mòdul de comunicacions es realitza utilitzant les anomenades comandes AT a través de la USART1. El funcionament és senzill: quan des del microcontrolador s'envia una comanda AT al mòdul de comunicacions, aquest retorna una resposta que pot ser la confirmació de que la comanda s'ha processat correctament (OK) o incorrectament (ERROR) i/o una cadena de caràcters amb la informació sol·licitada. Es mostren alguns exemples a la Taula 10.

Comanda enviada	Resposta	Descripció
AT+CGNSPWR=1	OK o ERROR	Retorna OK o ERROR segons si s'ha pogut habilitar o no el GPS del mòdul
AT+CSQ	+CSQ: 24,99 OK	Retorna la qualitat de la senyal i confirma la recepció
AT+CREG?	+CREG: 0,1 OK	Retorna si s'ha registrat correctament a la xarxa o no i confirma la recepció

Taula 10. Exemples de comandes AT

Com que s'han utilitzat més de 15 comandes AT s'ha considerat oportú crear una funció per facilitar la comunicació entre el microcontrolador i el mòdul de comunicacions, el diagrama de blocs del qual es mostra a la Figura 32.

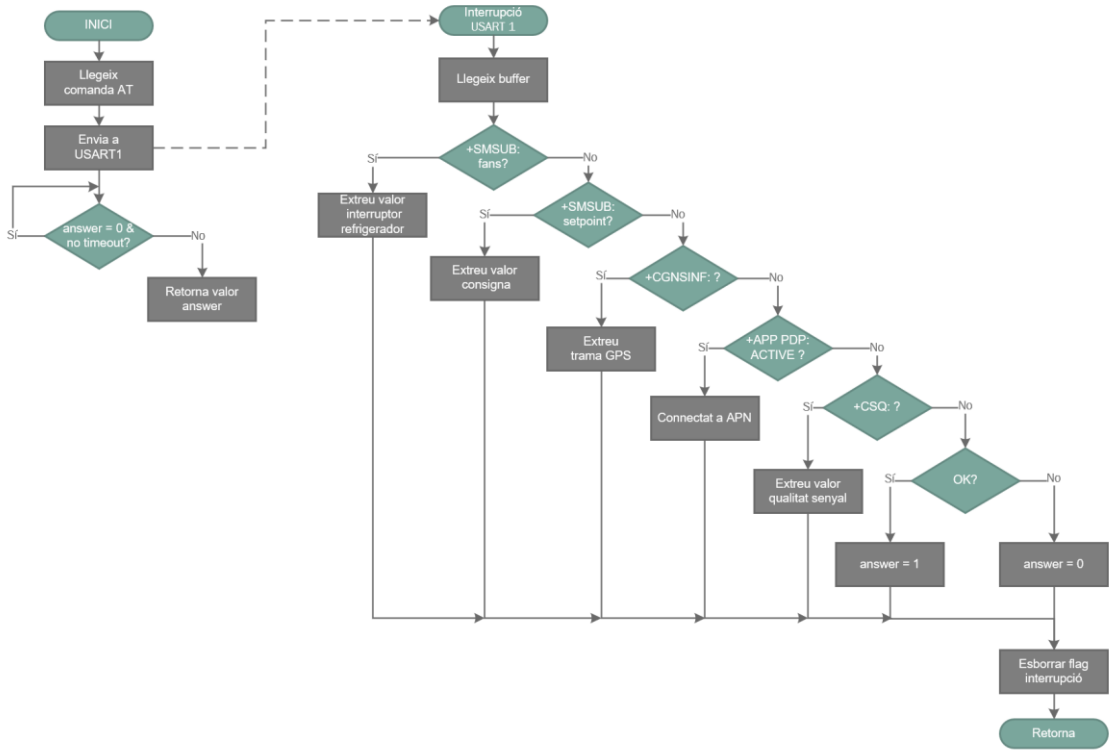


Figura 32. Diagrama de flux d'enviament/recepció de comandes AT
Elaboració pròpia

En aquesta funció el dispositiu llegeix la comanda a enviar al mòdul de comunicacions i l'envia a través de la USART1. Quan el mòdul de comunicacions envia la resposta cap al dispositiu, aquest activa la interrupció de la USART1 per llegir el buffer. En funció del missatge rebut, s'actua d'una manera o d'una altra. Per exemple, si es rep el missatge de subscripció de l'estat de l'interruptor de l'equip de refrigeració *(+SMSUB: "tfm/vehicles/v0/fans","1"*), s'extreu el valor "1" de la cadena de caràcters rebuda i es passa al programa principal per tal de determinar si cal encendre l'equip de refrigeració o no.

Mentrestant, el codi principal del programa es manté en espera d'obtenir la resposta de la interrupció. Mentre no s'obté resposta i no ha passat un temps prudencial (*timeout*), es segueix esperant. Si passat el temps d'espera no s'ha obtingut resposta o s'ha obtingut ERROR, se surt del bucle i s'actua en conseqüència, reintentant l'enviament de la comanda fins a tres vegades, com ja s'ha vist en el diagrama de blocs de la Figura 29.

Les comandes AT utilitzades en aquest projecte juntament amb la seva descripció i ús es poden trobar a l'annex *A.3 Comandes AT utilitzades* d'aquesta memòria. Per entrar en més detall o veure la resta de comandes disponibles per aquest xip, el propi fabricant, SIMCom, proporciona les fitxes tècniques on es descriuen totes les comandes AT compatibles i es proporcionen exemples d'ús. Les comandes AT genèriques pel xip SIM7000E s'han extret de [27]; les comandes referents a la implementació del protocol MQTT es troben a [28]; i les comandes necessàries per al protocol SSL estan a [29].

5. Nodes simulats

Com que només s'implementa un node físic, s'ha creat un simulador que permet generar nous vehicles i obtenir així una flota de vehicles que pugui ser visualitzada a la plataforma IoT. Així doncs, en aquest capítol es descriu el simulador implementat, com es generen els vehicles, d'on s'obtenen les dades d'interès d'aquests i finalment, com s'envien les dades cap a la plataforma.

5.1 El simulador de vehicles

Per realitzar la simulació d'aquests vehicles s'ha creat una aplicació en llenguatge *Python* on la part d'interfície gràfica s'ha desenvolupat utilitzant la biblioteca anomenada *Tkinker*. Aquesta es considera un estàndard de *Python* i és ràpida i fàcil d'implementar. La interfície gràfica creada per a aquest simulador es mostra a la Figura 33.

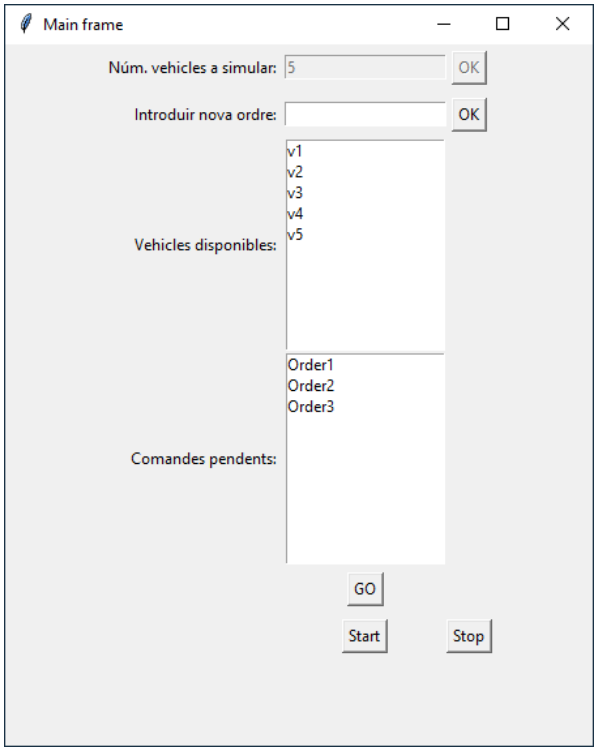


Figura 33. Interfície del simulador de vehicles
Elaboració pròpia

L'aplicació el que fa bàsicament és associar un vehicle amb una ordre determinada. Per exemple, el vehicle v1 se li pot assignar la Order5, al vehicle v2 se li pot assignar la Order2, i així successivament.

Per fer-ho, en primer lloc cal introduir el nombre de vehicles que es volen simular i prémer el botó OK. Això crea una llista de vehicles disponibles començant per v1 i fins a vn. El nombre de vehicles a simular no es podrà modificar fins que no es reinicialitzi el programa mitjançant el botó RESET.

Per altra banda, també es podran definir les ordres les quals s'hauran d'introduir una per una en el camp corresponent, respectant el format (Order1, Order2... Order8). Al prémer el botó OK, s'aniran acumulant a la llista d'ordres pendents. En aquest cas, sí que es podran anar introduir noves ordres mentre l'aplicació està en funcionament.

Un cop es disposa de la llista de vehicles i ordres, cal prémer el botó START per inicialitzar la plataforma IoT amb el nombre de vehicles establerts. Per altra banda, quan es desitgi finalitzar la simulació cal prémer el botó STOP per aturar la plataforma. Tot aquest procés d'inicialització i finalització de la plataforma IoT està explicat en més detall a l'annex *A.2 Configuració d'AWS EC2 i Lambda*.

Un cop la plataforma està inicialitzada, s'activa el botó GO que, a l'accionar-lo, permet associar el primer vehicle de la llista amb la primera ordre pendent, desapareixent de les llistes corresponents. Si es torna a pitjar el botó GO, s'associa el següent vehicle amb la següent ordre i així successivament fins que no hi hagi vehicles disponibles o ordres pendents. En aquest punt, no es poden atendre noves ordres i cal esperar a que hi hagi vehicles disponibles una altra vegada i s'hagin creat noves ordres. Els vehicles tornen a aparèixer a la llista quan hagin finalitzat l'ordre encomanada.

5.2 Les dades del simulador

Com es mostra a la Figura 34, les dades d'interès que s'envien a la plataforma IoT provenen de diverses fonts: algunes s'extreuen d'arxius CSV (valors separats per comes), d'altres de la pròpia plataforma i d'altres es generen de forma aleatòria o s'obtenen mitjançant una funció.

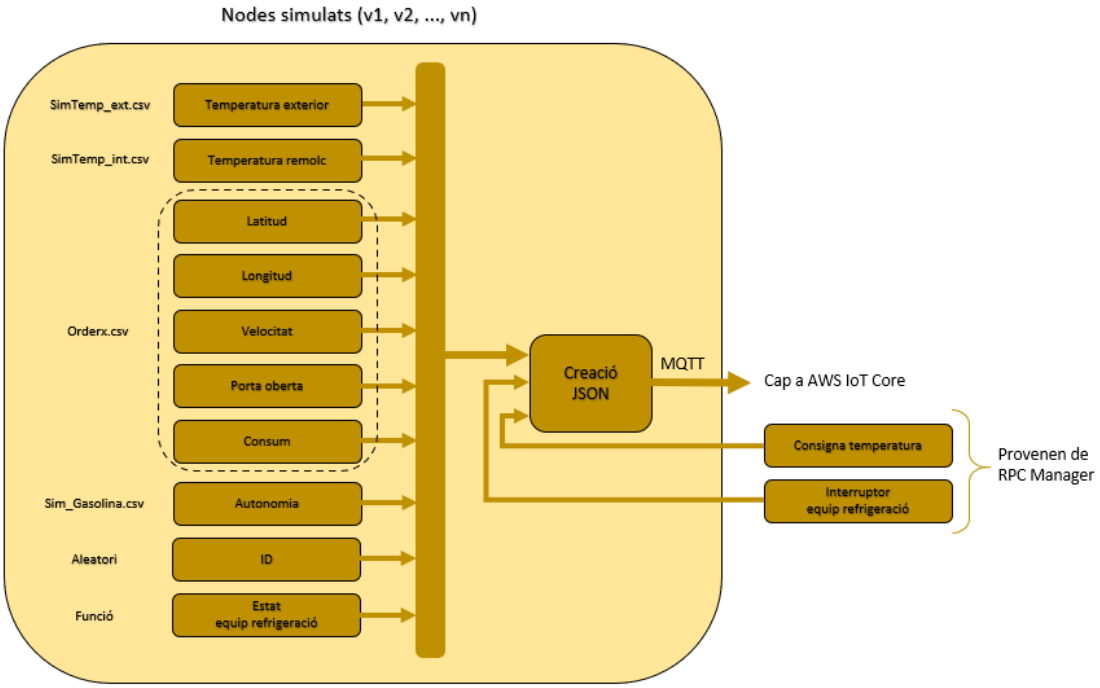


Figura 34. Principals elements del node simulat
Elaboració pròpia

Pel que fa a les ordres, s'ha creat un arxiu CSV per a cadascuna d'elles. Cada ordre porta associada la ruta que seguirà el vehicle (latitud i longitud), la velocitat, l'obertura de la porta i el consum. Aquestes dades s'han hagut de tractar conjuntament per tal d'evitar incongruències, com per exemple, que la velocitat sigui 0 mentre la latitud i la longitud van canviant de valor.

S'han creat un total de 8 rutes diferents. La zona per la qual circulen, els quilòmetres que recorre i el número de punts que té cadascuna d'elles es mostra a la Taula 11.

Ordre	Zona	núm. de punts	Km
Order1	Fornells	131	3.2
Order2	Girona centre	103	2.0
Order3	Montilivi	158	3.1
Order4	Salt	121	2.4
Order5	Santa Eugènia	180	3.6
Order6	Girona centre 2	168	3.3
Order7	Sant Gregori	209	4.2
Order8	Vilablareix	204	4.0

Taula 11. Rutes generades

Les rutes s'han creat mitjançant una eina online anomenada *NMEA Generator* [49], la qual permet dibuixar la ruta sobre un mapa podent escollir la distància entre els punts (la qual cosa defineix la velocitat del vehicle) i l'hora d'inici de la ruta, com es mostra a la Figura 35. Un cop dibuixada permet exportar les coordenades de la ruta en format NMEA o CSV. Per a aquest projecte, les dades s'exportaran a CSV ja que per la resta de paràmetres també s'ha utilitzat aquest format.

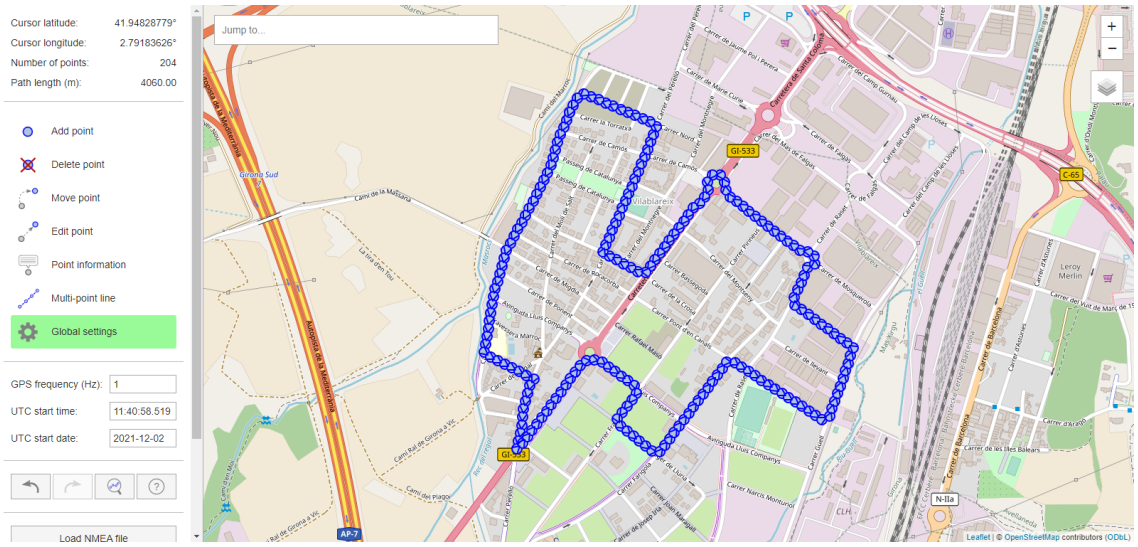
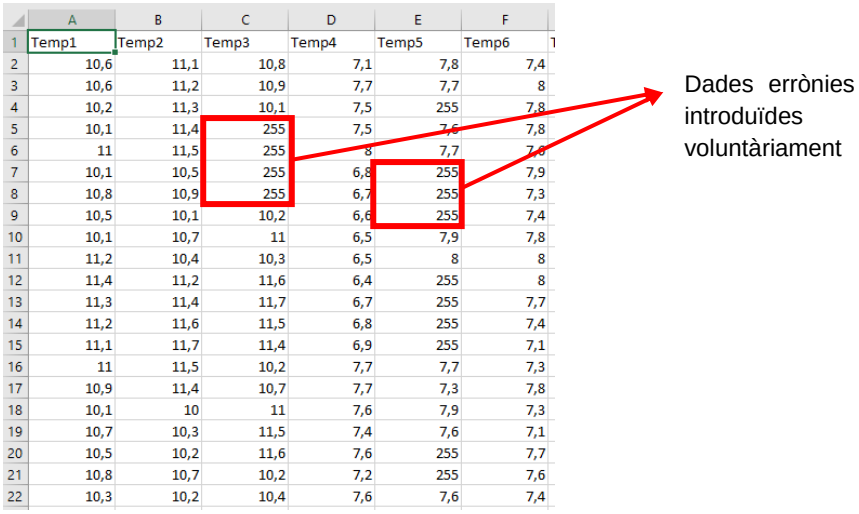


Figura 35. Ruta dibuixada amb NMEA Generator
Elaboració pròpia

Pel que fa a la resta de dades que s'extreuen d'arxius CSV (temperatura exterior, la temperatura del remolc i l'autonomia) es creen diverses columnes de valors, com es mostra a la Figura 36. Cada vegada que s'associa un vehicle amb una ordre, s'escull una columna de forma aleatòria d'on extreure les dades per tal d'oferir més variabilitat a la simulació.

Com es pot observar, també s'han introduït dades errònies per comprovar i demostrar que el sistema és capaç de detectar-les i tractar-les.



	A	B	C	D	E	F
1	Temp1	Temp2	Temp3	Temp4	Temp5	Temp6
2	10,6	11,1	10,8	7,1	7,8	7,4
3	10,6	11,2	10,9	7,7	7,7	8
4	10,2	11,3	10,1	7,5	255	7,8
5	10,1	11,4	255	7,5	7,6	7,8
6	11	11,5	255	8	7,7	7,6
7	10,1	10,5	255	6,8	255	7,9
8	10,8	10,9	255	6,7	255	7,3
9	10,5	10,1	10,2	6,6	255	7,4
10	10,1	10,7	11	6,5	7,9	7,8
11	11,2	10,4	10,3	6,5	8	8
12	11,4	11,2	11,6	6,4	255	8
13	11,3	11,4	11,7	6,7	255	7,7
14	11,2	11,6	11,5	6,8	255	7,4
15	11,1	11,7	11,4	6,9	255	7,1
16	11	11,5	10,2	7,7	7,7	7,3
17	10,9	11,4	10,7	7,7	7,3	7,8
18	10,1	10	11	7,6	7,9	7,3
19	10,7	10,3	11,5	7,4	7,6	7,1
20	10,5	10,2	11,6	7,6	255	7,7
21	10,8	10,7	10,2	7,2	255	7,6
22	10,3	10,2	10,4	7,6	7,6	7,4

Figura 36. Dades simulades de la temperatura exterior del vehicle
Elaboració pròpia

Pel que fa als valors de la consigna de temperatura i de l'interruptor de l'equip refrigerador, aquests són enviats des de la pròpia plataforma IoT. Aquests valors són inicialment precarregats des dels arxius de text anomenats: *default values temp.txt* i *default values fan.txt* ubicats dins la instància EC2 i són els que es mostren en els ginys de control quan s'obre el quadre de comandament del vehicle corresponent de la plataforma IoT.

A partir d'aquí, qualsevol modificació que es faci d'aquests valors mitjançant els ginys del quadre de comandament, queden guardats en aquests arxius de text i són transmesos cap al vehicle simulat corresponent.

Amb aquests valors rebuts es podrà determinar el valor del paràmetre *Estat de l'equip de refrigeració*, de forma similar a com es fa en el node físic: es posarà en marxa quan la consigna de temperatura estigui per sota la temperatura de l'interior del remolc sempre que l'interruptor de l'equip de refrigeració estigui activat i s'aturarà quan la temperatura de l'interior del remolc estigui per sota de la consigna de temperatura o l'interruptor estigui desactivat.

Finament, s'ha creat el paràmetre *id* per identificar de forma unívoca cadascun dels trajectes realitzats pels vehicles. La codificació utilitzada per generar aquest identificador és el que es mostra a la Figura 37. La part aleatòria és generada a través de la funció *uuid4()* de *Python* la qual crea un UUID aleatori (*Universally Unique Identifier*) de la qual s'agafaran els primers 6 dígits. Serà necessari importar el mòdul uuid mitjançant la instrucció *import uuid* per a que funcioni.

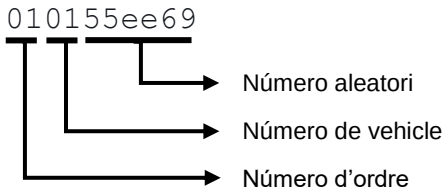


Figura 37. Identificador únic de trajecte
Elaboració pròpia

5.3 Comunicació amb la plataforma IoT

Cada vegada que es genera una associació vehicle-ordre des de la interfície del simulador, es crea un fil d'execució en paral·lel i dins un bucle es van extraient els paràmetres fila per fila de cadascun dels arxius CSV definits anteriorment. Amb aquestes valors es forma un missatge (*payload*) amb el format JSON, com el mostrat a la Figura 38, i s'envia de forma periòdica a la plataforma IoT mitjançant la publicació d'aquest missatge al tòpic del vehicle corresponent. Un cop s'arriba a l'última fila de l'arxiu *Orderx.csv*, que és el que determina la longitud del trajecte, l'ordre es destrueix i el vehicle torna a aparèixer a la llista de vehicles disponibles.


```
{
  "temp_int": 7.7,
  "temp_ext": 15.4,
  "latitude": 41.93265524047908,
  "longitude": 2.810910478662444,
  "speed": 56,
  "openDoor": 0,
  "setpoint": "11",
  "fans": "1",
  "id": "010155ee69",
  "gasol": 751
}
```


Figura 38. *Payload* en format JSON enviat a la plataforma IoT
Elaboració pròpia

Per enviar aquestes dades cap a la plataforma IoT mitjançant el protocol MQTT, *Python* disposa d'una biblioteca de client MQTT anomenada *Paho*, que cal importar en el codi del programa. Aquesta proporciona una classe client que permet que les aplicacions es connectin a un broker MQTT per publicar missatges i subscriure's a tòpics.

A la Figura 39, es mostren les línies de codi necessàries per definir i poder establir la comunicació amb el broker d'AWS IoT Core. De la mateixa manera que el node físic, també és necessari obtenir els certificats i les claus corresponents per connectar-se amb el servei. Tanmateix, per no complicar excessivament el codi, s'utilitzen els mateixos certificats i claus per a tots els vehicles simulats.

```
broker = "a1jdouxxxxxxxx-ats.iot.us-west-2.amazonaws.com"
broker_port = 8883
mqtt_keep_alive = 60
mqtt_publish_topic = "tfm/vehicles"
client = mqtt.Client(client_id=clientID, protocol=mqtt.MQTTv31)
client.on_connect = on_connect
client.on_message = on_message
client.on_publish = on_publish
client.tls_set(ca_certs="rootCA.pem", certfile="cert.crt" , keyfile="private.key")
client.connect(broker, broker_port, mqtt_keep_alive)
client.loop_start()
```

Figura 39. Codi per connectar amb el broker d'AWS IoT Core
Elaboració pròpia

Com es pot veure a les anteriors línies de codi, es defineix la direcció del broker, el port i el temps actiu (màxim temps que pot transcorre entre el moment en que un client acaba de transmetre un paquet de control i quan comença a enviar el següent). Per defecte, s'utilitza el port 8883 per MQTT sobre TLS, i el port 1883 quan la transmissió és sense xifrar. Per altra banda, es defineix el tòpic, el client, els certificats i claus per treballar amb TLS i els callbacks de connexió, missatge i publicació.

6. Plataforma Cloud

En aquest capítol es descriu la plataforma al núvol desenvolupada sobre Amazon Web Services (AWS). Per una banda, permetrà gestionar i presentar la informació dels vehicles en temps real mitjançant la plataforma IoT Thingsboard i, per l'altre, realitzar un anàlisi de la informació històrica registrada mitjançant les eines que disposa el propi AWS, com son AWS IoT Analytics i Amazon QuickSight.

Per a la presentació i gestió de dades en temps real s'ha escollit Thingsboard per ser una plataforma específicament dissenyada per IoT, molt intuïtiva i ràpida a l'hora de desenvolupar quadres de comandament en temps real. A més, disposa de grans possibilitats de personalització i permet no només visualitzar les dades enviades pels dispositius sinó també enviar paràmetres cap als dispositius. No obstant, no disposa d'eines per fer l'anàlisi forense de les dades.

En canvi, la plataforma d'AWS és una plataforma molt potent per a l'anàlisi i tractament de la informació històrica ja que fins i tot disposa d'eines avançades d'intel·ligència artificial i aprenentatge automàtic (*Amazon SageMaker*), però no disposa de serveis específics en temps real pel seguiment de vehicles, com pot ser la visualització de la ubicació del vehicle sobre un mapa o la possibilitat d'enviar ordres als vehicles.

És per això que s'ha escollit el millor de cada plataforma per obtenir una solució IoT en el núvol completament integrada dins d'AWS.

6.1 Gestió i presentació d'informació en temps real

En aquest apartat es presenta el bloc encarregat d'interactuar mitjançant una plataforma IoT en temps real tant amb el node físic com amb els diferents nodes simulats. Per tant, compleix una doble funció: recollir les dades enviades des d'AWS IoT provinents dels nodes per presentar-les en una interfície gràfica en temps real; i, a la vegada, des d'aquesta mateixa interfície enviar ordres als nodes per controlar el seu estat.

Aquest bloc s'ha desplegat en un servidor virtual Linux mitjançant el servei d'Amazon EC2 on s'ha instal·lat la plataforma IoT Thingsboard i s'ha desenvolupat el programa que permet enviar les ordres cap als nodes mitjançant un model petició/resposta.

Després d'analitzar les diferents alternatives, s'ha escollit Thingsboard pel fet que es tracta d'una plataforma IoT de codi obert que permet un ràpid desenvolupament, gestió i escalat de projectes IoT i pot ser instal·lada tant en local com en el núvol. El fet que es trobi disponible a AWS Marketplace facilita la integració amb AWS, ja que seguint uns senzills passos de configuració, es crea una instància EC2 i s'hi instal·la de forma automàtica l'aplicació.

La utilització d'aquestes màquines virtuals suposen un cost en funció del tipus d'instància EC2 escollida i per cada hora d'ús. Per no tenir el programari executant-se de forma continua i minimitzar així costos s'ha considerat oportú iniciar-la de forma automàtica només en aquells moments en què s'utilitzi i apagar-la quan es deixi d'utilitzar.

Els detalls de la creació de la instància, la configuració inicial de Thingsboard i l'encès i apagat automatitzat de la instància s'expliquen a l'annex *A.2 Configuració d'AWS EC2 i Lambda* d'aquesta memòria.

Thingsboard és una de les poques plataformes IoT que s'han trobat que, a banda d'enviar les dades dels dispositius cap a la plataforma IoT, com la majoria de plataformes realitzen, també disposa de ginys que permeten la comunicació a la inversa, és a dir, enviar ordres des de la pròpia plataforma IoT cap als dispositius. Això és el que permet controlar l'engegat/apagat de l'equip refrigerador o la modificació de la temperatura de consigna.

Altres característiques destacades per les quals s'ha escollit Thingsboard com a plataforma IoT en temps real són les que es presenten a continuació:

- Aprovisionament de dispositius, actius i clients i definició de relacions entre ells.
- Recopilació i visualització de dades de dispositius i actius.
- Anàlisis de la telemetria entrant i activació d'alarmes amb processament d'esdeveniments complexes.
- Control de dispositius mitjançant les crides a procediments remots (RPC).
- Disseny de quadres de control dinàmics i receptius amb presentació d'informació i telemetria de dispositius o actius als clients.
- Habilitació de funcions específiques de casos d'ús mitjançant cadenes de regles personalitzades.
- Enviament de dades dels dispositius a altres sistemes.

6.1.1 Connexió AWS IoT - Thingsboard

Per tal de poder representar les dades als quadres de comandaments creats a Thingsboard cal realitzar un parell de passos previs: establir la connexió amb el broker d'AWS IoT i adaptar les dades al format de Thingsboard. L'esquema que segueixen les dades des dels dispositius fins al *dashboard* en temps real és el que es mostra a la Figura 40.

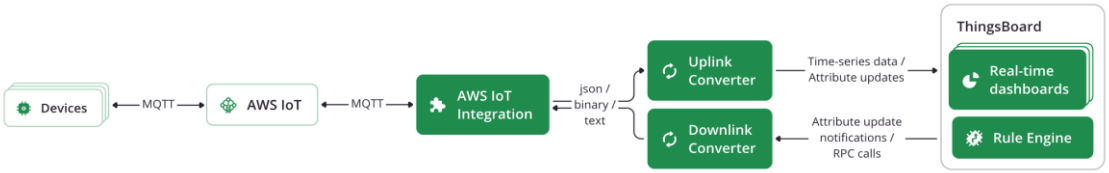


Figura 40. Connexió AWS IoT – Thingsboard
Extret de thingsboard.io

Thingsboard permet la integració amb diferents serveis al núvol i un d'ells, és el d'AWS IoT. Així, el bloc d'*AWS IoT Integration* actua com un client MQTT, de manera que subscrivint-se al tòpic *tfm/vehicles/+,* el broker d'AWS IoT reenvia tots els missatges publicats pels dispositius a Thingsboard. Per fer-ho, cal crear una nova integració on s'ha de configurar la URL del broker, les credencials d'AWS IoT i el tòpic al qual subscriure's. Per obtenir les credencials (claus i certificats) cal seguir el mateix procés que l'explicat a l'annex *A.1 Configuració d'AWS IoT Core* d'aquesta memòria.

A continuació, cal configurar el bloc de *Uplink Converter*, el qual és l'encarregat d'adaptar les dades enviades pels dispositius al format de Thingsboad. Per realitzar això, Thingsboard disposa d'una funció-plantilla que permet convertir les dades que provenen del bloc d'AWS IoT Integration en format JSON amb sèries temporals de dades que són les que es passaran al quadre de comandament de temps real.

Finalment cal crear els diferents vehicles que formen la flota amb els mateixos noms que s'han definit el nodes físic i els nodes simulats. En aquest cas, com la llicència de Thingsboard adquirida només admet la creació de 10 vehicles, es creen 10 dispositius que s'anomenaran v0, v1, v2, v3... v9 en què v0 correspon al node físic i la resta als nodes simulats.

Per tant, els passos a seguir per tal que Thingsboard representi les dades enviades pels dispositius són els que es mostren a la Figura 41.

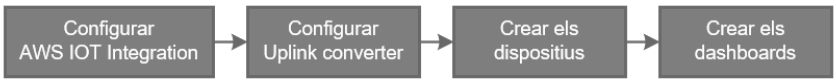


Figura 41. Passos per configurar Thingsboard
Elaboració pròpia

6.1.2 Quadres de comandament (*dashboards*)

Per visualitzar les dades enviades pels dispositius (tant del node físic com dels nodes simulats) en temps real, s'ha creat un quadre de comandament principal on es mostra l'estat general de la flota i quadres de comandament secundaris on es mostra la informació pròpia de cada vehicle.

En el quadre de comandament principal es poden visualitzar les ubicacions dels vehicles sobre un mapa, les variables més importants d'aquests (temperatura interior del remolc, consigna de temperatura, velocitat del vehicle, estat de la porta del remolc...) en una taula i les alarmes que s'han activat tal i com es mostra a la Figura 42.

Clicant sobre el vehicle des de la taula de variables o des del propi mapa, s'accedeix als panells propis de cada dispositiu per veure l'estat de cadascun d'ells en detall.

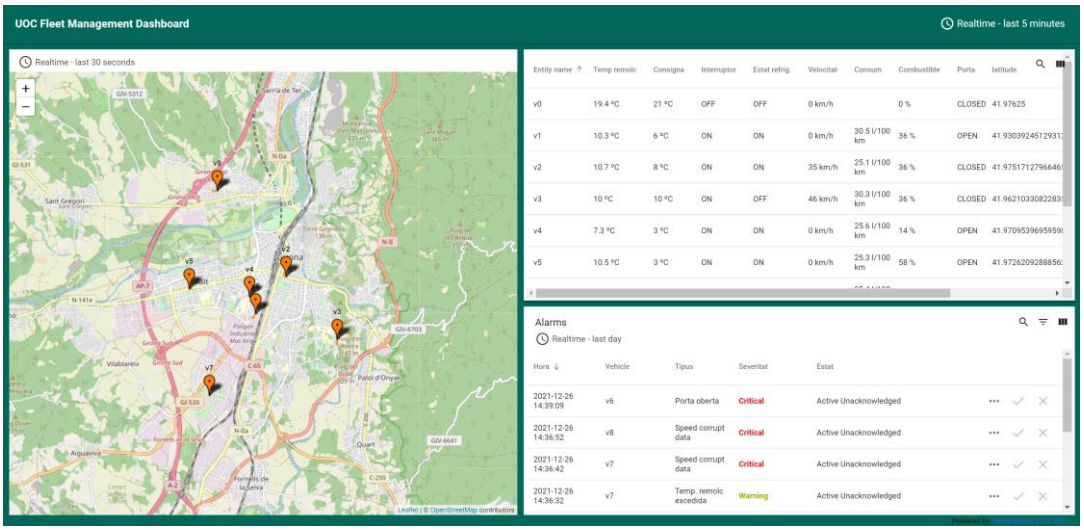


Figura 42. Dashboard principal
Elaboració pròpia

Pels quadres de comandament secundaris, s'ha utilitzat una funcionalitat molt útil que disposa Thingsboard anomenada *Dashboard state*, la qual permet crear un mateix *dashboard* per a tots els vehicles però amb les dades corresponents a cadascun d'ells segons el dispositiu seleccionat. Això permet que el sistema sigui altament escalable, ja que independent del nombre de vehicles que tingui la flota, amb la creació d'un sol *dashboard* es poden visualitzar les dades pròpies de tots dispositius.

La informació particular que es mostra de cada vehicle és l'identificador de trajecte, el trajecte seguit pel vehicle sobre un mapa; la velocitat actual del vehicle i una gràfica de la seva evolució; la temperatura exterior actual, la temperatura del remolc actual i la consigna seleccionada juntament amb una gràfica de la seva evolució; el consum actual i una taula amb les alarmes actives.

A més, també disposa de ginys de control com un interruptor i un control de perilla que permeten activar/desactivar l'equip de refrigeració i escollir la temperatura de consigna, respectivament. El funcionament d'aquests ginys i com envien/reben la informació s'explica en més detall a l'apartat *6.1.4 RPC Manager.*

Per altra banda, des de la part superior dreta tant del quadre de comandament principal com dels secundaris es disposa d'un menú desplegable que permet seleccionar una finestra de temps. D'aquesta manera, es pot revisar l'històric de trajectes i de dades rebudes en un interval de temps seleccionat o d'una data específica.

A la Figura 43 es pot veure el quadre de comandament secundari, en aquest cas, del vehicle 3.

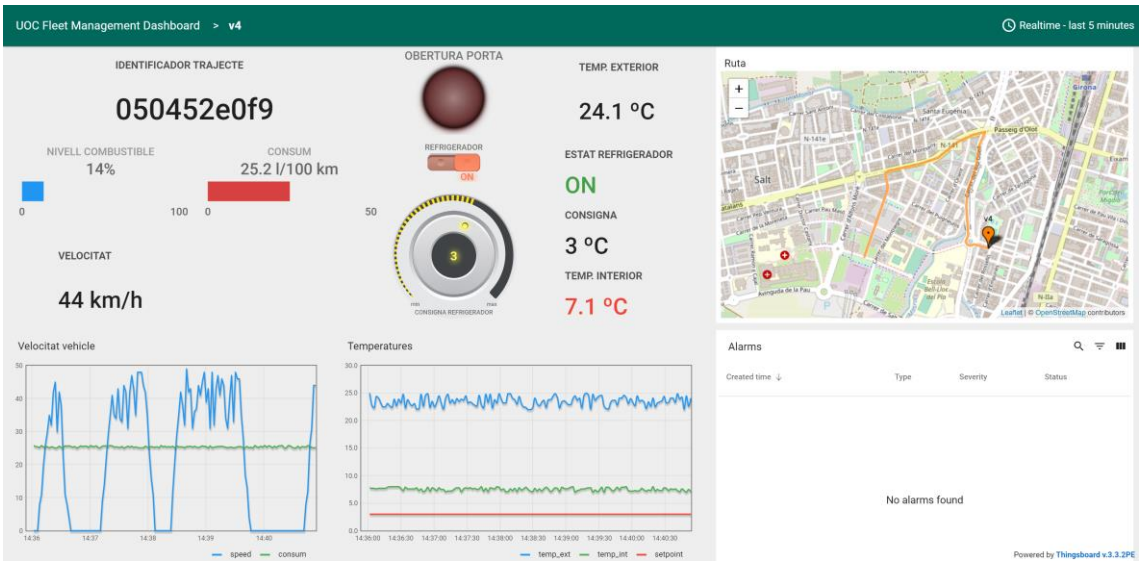


Figura 43. Dashboard del vehicle v3
Elaboració pròpia

6.1.3 Gestió d'alarmes

Thingsboard també proporciona la capacitat de crear i administrar alarmes relacionades amb els dispositius i mostrar-les en una taula en el dashboard. Es poden definir les condicions d'activació/desactivació de l'alarma, la seva gravetat, el període d'activació i els detalls de la seva activació/desactivació.

Les alarmes creades són comunes per a tots els dispositius d'un mateix perfil, de manera que creant diferents tipus de perfils es podrien crear condicions particulars per diferents grups de vehicles.

En el cas d'aquest projecte, s'ha creat un únic perfil que engloba a tots els vehicles de la flota. Algunes de les alarmes creades per provar el correcte funcionament són les que es presenten a la Taula 12. Evidentment, el tipus l'alarma i les condicions d'activació depenen de cada aplicació d'ús específica.

Nom alarma	Severitat	Condicions	Tipus condició	Duració
Temp remolc alta	Warning	Temp_int > setpoint	Duració	30 s.
	Critical	Temp_int > setpoint	Duració	60 s.
Temp remolc baixa	Warning	Temp_int < setpoint	Duració	30 s.
	Critical	Temp_int < setpoint	Duració	60 s.
Porta oberta	Info	openDoor = 1 and speed = 0	Duració	60 s.
	Warning	openDoor = 1 and speed ≠ 0	Simple	-
Velocitat	Warning	Speed > 100	Simple	-

Taula 12. Taula d'alarmes

Thingsboard disposa d'un giny específic per les alarmes (*Alarm widget*) en forma de taula on es pot visualitzar l'hora en què s'ha disparat l'alarma, de quin vehicle prové, el tipus d'alarma i la seva gravetat. També disposa del camp *Status* en què es pot indicar si l'alarma ha estat assabentada i/o esborrada.

Cal tenir en compte també que de la manera que Thingboard gestiona les alarmes, en un mateix moment, només hi haurà una alarma activa amb el mateix originador, tipus i hora d'inici. Així doncs, cada vegada que es genera una nova alarma amb aquestes condicions, l'únic que varia és el valor que indica l'última vegada que s'ha activat aquesta alarma.

6.1.4 RPC Manager

Com ja s'ha comentat, una de les característiques destacades de Thingsboard, és que permet enviar comandes cap als dispositius i des dels dispositius i rebre els resultats de l'execució mitjançant les anomenades *Remote Procedure Calls* (RPC), de la manera que es mostra a la Figura 44.

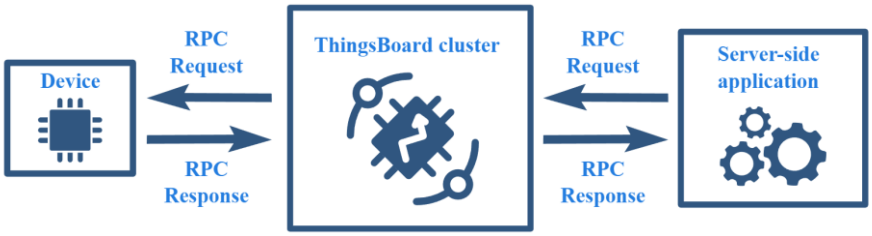


Figura 44. RPC des del servidor bidireccional
Extret de thingsboard.io

Aquestes RPC són generades pels ginys de control que s'han vist anteriorment com l'interruptor o el control de perilla. Quan s'obre un dashboard en el qual hi ha un d'aquests ginys, es realitza un petició automàtica anomenada *getValue()* i el giny s'actualitza amb el valor rebut del dispositiu. Per altra banda, quan es modifica el valor del giny, per exemple, un interruptor passa de ON a OFF o el control de perilla canvia de valor, es realitza una altra petició del tipus *setValue()* per actualitzar el valor del dispositiu.

L'inconvenient és que Thingsboard obliga a utilitzar els seus propis tòpics en què, a més, no s'hi especifica quin vehicle ha fet aquesta petició. Això impossibilita que aquestes dades obtingudes des de la plataforma IoT puguin ser enviades directament al broker d'AWS.

Per aquesta raó, ha estat necessari implementar el bloc *RPC Manager* el qual s'encarrega de gestionar aquestes peticions, determinant quin dispositiu realitza la petició i reenviant-la al servei d'AWS IoT Core. Per fer-ho, cal implementar el propi broker de Thingsboard i així poder publicar i subscriure's als tòpics requerits.

Per subscriure's a les ordres RPC des del servidor, s'ha d'enviar el missatge SUBSCRIBE al tòpic següent:

```
v1/devices/me/rpc/request/+
```

Una vegada subscrit, el client rep les comandes individuals com a missatge PUBLISH al tòpic corresponent:

```
v1/devices/me/rpc/request/$request_id
```

on el *$request_id* és un identificador de sol·licitud del tipus enter.

El client a continuació, ha de publicar la resposta al següent tòpic:

```
v1/devices/me/rpc/response/$request_id
```

Gràcies a la creació de fils (*threads*) per a cadascun dels dispositius (tant pel físic com pels simulats) es pot determinar de quin dispositiu prové l'ordre RPC i així actuar sobre els seus valors específics.

En les línies de la Figura 45 es mostra un petit extracte del codi que possibilita la modificació de la temperatura de consigna del remolc mitjançant el control de perilla:

```
rpcValue = int(threading.current_thread().name)
if msg.topic.startswith('v1/devices/me/rpc/request/'):
        requestId = msg.topic[len('v1/devices/me/rpc/request/'):len(msg.topic)]
        data = json.loads(msg.payload)

        if data['method'] == 'getValue':
            client2.publish('v1/devices/me/rpc/response/'+requestId, json.dumps(int(vehiclesTemp[rpcValue].get())), 1)
            client3.publish("tfm/vehicles/v" + threading.current_thread().name + "/setpoint", json.dumps(int(vehiclesTemp
            [rpcValue].get())), 1)


          if data['method'] == 'setValue':
             params = data['params']
             vehiclesTemp[rpcValue].set(str(params))
             client3.publish("tfm/vehicles/v" + threading.current_thread().name + "/setpoint", json.dumps(int(params)), 1)
```

Figura 45. Codi RPC per a la consigna de temperatura del remolc
Elaboració pròpia

Així doncs, si es rep un tòpic que coincideix amb una petició RPC del tipus *getValue*, es publica la resposta al tòpic que indica Thingsboard amb la id corresponent i el missatge proporcionat per l'RPC Manager, que en aquest cas serà la consigna de temperatura. A la vegada, a través d'un altre client MQTT (client3) també es publica aquest mateix missatge a AWS IoT al tòpic corresponent.

En canvi, si es rep la petició RPC del tipus *setValue*, s'agafa el valor que marca el giny, en aquest cas el control de perilla, s'actualitza el valor a l'RPC Manager i, com abans, es publica aquest mateix valor a AWS IoT al tòpic corresponent.

Amb la implementació d'aquest bloc, s'aconsegueix aquesta comunicació bidireccional entre el dispositiu i la plataforma Thingsboard.

6.2. Anàlisi d'informació històrica registrada

En aquest capítol es descriuen els serveis d'AWS utilitzats per poder realitzar un anàlisi forense de les dades enviades pels dispositius IoT i presentar-ho d'una manera gràfica i intuïtiva per així amb un simple cop d'ull tenir una visió global de la flota i obtenir informació valuosa per prendre accions i decisions en la gestió de la flota.

Per aconseguir-ho, s'ha utilitzat AWS IoT Analytics que permet recopilar, preprocessar i emmagatzemar les dades dels dispositius IoT redirigides des del servei d'AWS IoT Core; i Amazon Quicksight que permet analitzar, agrupar i filtrar les dades per a que puguin ser visualitzades mitjançant la utilització de gràfics circulars, gràfics de barres, taules o indicadors.

Les funcions que s'han implementat per realitzar l'anàlisi de la flota de vehicles són les següents:

- Número de vehicles actius
- Número de dies que un vehicle porta aturat
- Número de trajectes totals i per vehicle
- Distàncies recorregudes totals i per vehicle
- Mitjana de quilòmetres totals per trajecte i per vehicle
- Temperatura màxima del remolc registrada i per vehicle
- Temperatura mitjana del remolc registrada i per vehicle
- Percentatge del trajecte on la temperatura està per sobre la consigna i contribució de cada vehicle
- Núm. de parades totals efectuades i per vehicle
- Percentatge del trajecte on el vehicle està aturat i contribució de cada vehicle
- Percentatge del trajecte on el vehicle té la porta oberta i contribució de cada vehicle
- Velocitat màxima registrada dels vehicles i per vehicle
- Velocitat mitjana dels vehicles i per vehicle
- Dades errònies (fora de rang)

6.2.1 AWS IoT Analytics

Les eines tradicionals d'anàlisis i intel·ligència empresarial estan dissenyades per processar dades estructurades. Les dades sense processar d'IoT provenen habitualment de dispositius que registren dades menys estructurades (com temperatura, moviment o so). Com a resultat, les dades d'aquests dispositius poden tenir missatges corruptes o lectures falses que han de netejar-se abans de que es pugui realitzar l'anàlisi.

AWS IoT Analytics permet tractar aquests problemes i automatitzar els passos necessaris per analitzar les dades dels dispositius IoT. AWS IoT Analytics és capaç de filtrar, transformar i enriquir les dades d'IoT abans d'emmagatzemar-los per al seu anàlisi.

Mitjançant la creació d'una regla a AWS IoT Core es defineix que totes les dades que arribin a AWS IoT Core siguin reenviades a AWS IoT Analytics. A IoT Analytics és necessari crear el que s'anomena un canal, un *pipeline*, un *datastore* i un *dataset*.

El canal (*Channel*) és l'encarregat de recopilar les dades enviades des de AWS IoT Core i arxivar els missatges sense processar abans de publicar-los en una pipeline. Aquests missatges sense processar, s'emmagatzemen a Amazon S3 (*Amazon Simple Storage Service*) que pot ser administrat per un mateix o per AWS IoT Analytics.

El *Pipeline* és l'encarregat d'ingerir els missatges del canal i processar-los abans d'emmagatzemar-los en el magatzem de dades (*datastore*). De la mateixa manera que amb els missatges del canal sense processar, els missatges processats del magatzem s'emmagatzemen a Amazon S3 que pot ser administrat per un mateix o per AWS IoT Analytics. Finalment, amb les dades del *datastore* es creen diferents *datasets* a partir de la declaració d'una sentència SQL.

Per a aquest projecte, per obtenir certs paràmetres a partir de les dades enviades pels dispositius, ha estat necessari crear diferents *datasets*. Per exemple, per a tot allò relacionat amb l'obertura de la porta i la velocitat del vehicle, s'ha creat el *dataset* anomenat *uocfleet_0speed_and_opendoor_detection*, mitjançant el qual es poden obtenir algunes de les funcions presentades a l'inici d'aquest capítol com el número de parades totals efectuades o el percentatge del trajecte on els vehicles tenen la porta oberta.

La sentència SQL creada per a obtenir el *dataset* comentat, es mostra a la Figura 46.

```
SELECT
            w2.vehicleid,
            w2.id,
            w2.ts,
            w2.speed,
            lead(w2.speed) OVER (order by w2.vehicleid, w2.ts ASC) as next_speed,
            w2.openDoor,
            lead(w2.openDoor) OVER (order by w2.vehicleid, w2.ts ASC) as
next_openDoor
        FROM
            uocfleetdata_datastore w2
        ORDER BY
            w2.vehicleid, w2.ts ASC
```

Figura 46. Sentència SQL de detecció de porta oberta i vehicle aturat
Elaboració pròpia

Els datasets que han estat necessaris crear per poder obtenir totes les funcions presentades a l'inici d'aquest capítol, es mostren a la Taula 13.

Nom dataset	Funció
uocfleetdata_dataset	Obtenir dades relacionades amb la tempertura (núm. de vegades que la temperatura ha superat la consigna) i extreure dades errònies o fora de rang
uocfleet_distance	Obtenir distància recorreguda, duració de la ruta, hora d'inici i finalització de la ruta
uocfleet_0speed_and_opendoor_detection	Obtenir dades relacionades amb l'obertura del remolc (núm. d'obertures de porta, durada de les obertures...) i la velocitat del vehicle (núm. de parades, durada de les parades)

Taula 13. Datasets creats

6.2.2 Amazon Quicksight

Amazon Quicksight és un servei d'intel·ligència empresarial (BI) en el núvol per crear visualitzacions, realitzar anàlisis ad hoc i obtenir ràpidament informació empresarial a partir de les dades emmagatzemades. *Amazon Quicksight* permet importar les dades de diferents fonts de dades d'AWS amb un rendiment de consultes ràpid i receptiu mitjançant l'ús d'un motor de memòria sòlid anomenat SPICE.

Gràcies al seu entorn visual interactiu accessible des de qualsevol dispositiu de xarxa i des de dispositiu mòbils, permet explorar i interpretar la informació per poder prendre decisions de forma ràpida i efectiva.

Per a aquest projecte, s'han importat els tres *datasets* creats a l'apartat anterior. Quicksight, de forma automàtica i per a cadascun dels datasets, extreu el nom de les variables a partir de les quals també es poden implementar noves variables, com per exemple, una variable que permeti conèixer quantes vegades la temperatura interior del remolc ha estat per sobre la consigna.

Algunes d'aquestes noves variables creades a partir de les variables originals es mostren a la Taula 14.

Nova variable	Fórmula	Funció
temps_aturat	dateDiff(max(ts),now())	Mostra el nombre de dies que un vehicle porta aturat a partir de l'últim registre obtingut
mitjana_km	sum({dist_km})/distinct_count({id})	Obté la mitjana de quilòmetres realitzats per trajecte
duration_stop	countIf({id}, {speed}=0)/count({id})	Obté el tant per 1 de registres on la velocitat val 0.
duration_opened_door	countIf({id}, {openDoor}=1)/count({id})	Obté el tant per 1 de registres on la porta està oberta.

Taula 14. Noves variables creades

Finalment, amb la combinació de les variables originals i les variables creades, es poden crear els anàlisis que es requereixin i obtenir un quadre de comandament que reflecteixi els resultats. A la

Figura 47, es mostra el primer *dashboard* creat per a la solució IoT plantejada en aquest projecte. Es poden visualitzar tots els paràmetres llistats a l'inici d'aquest capítol com són el nombre de vehicles actius, el número de dies que un vehicle porta aturat, les distàncies recorregudes per cada vehicle, el percentatge del trajecte on la temperatura ha estat per sobre la consigna, entre d'altres. A més, totes aquestes dades, poden ser filtrades per vehicle i per data o període de temps mitjançant els controls de la part superior del *dashboard*.

Per altra banda, es disposa d'un segon *dashboard*, mostrat a la Figura 48 al qual s'accedeix a través de les pestanyes de la part superior, on es poden analitzar les tendències a nivell temporal d'alguns dels paràmetres anteriors com el número de trajectes realitzats per vehicle, el número de quilòmetres realitzats, el número de parades i la temperatura mitjana.

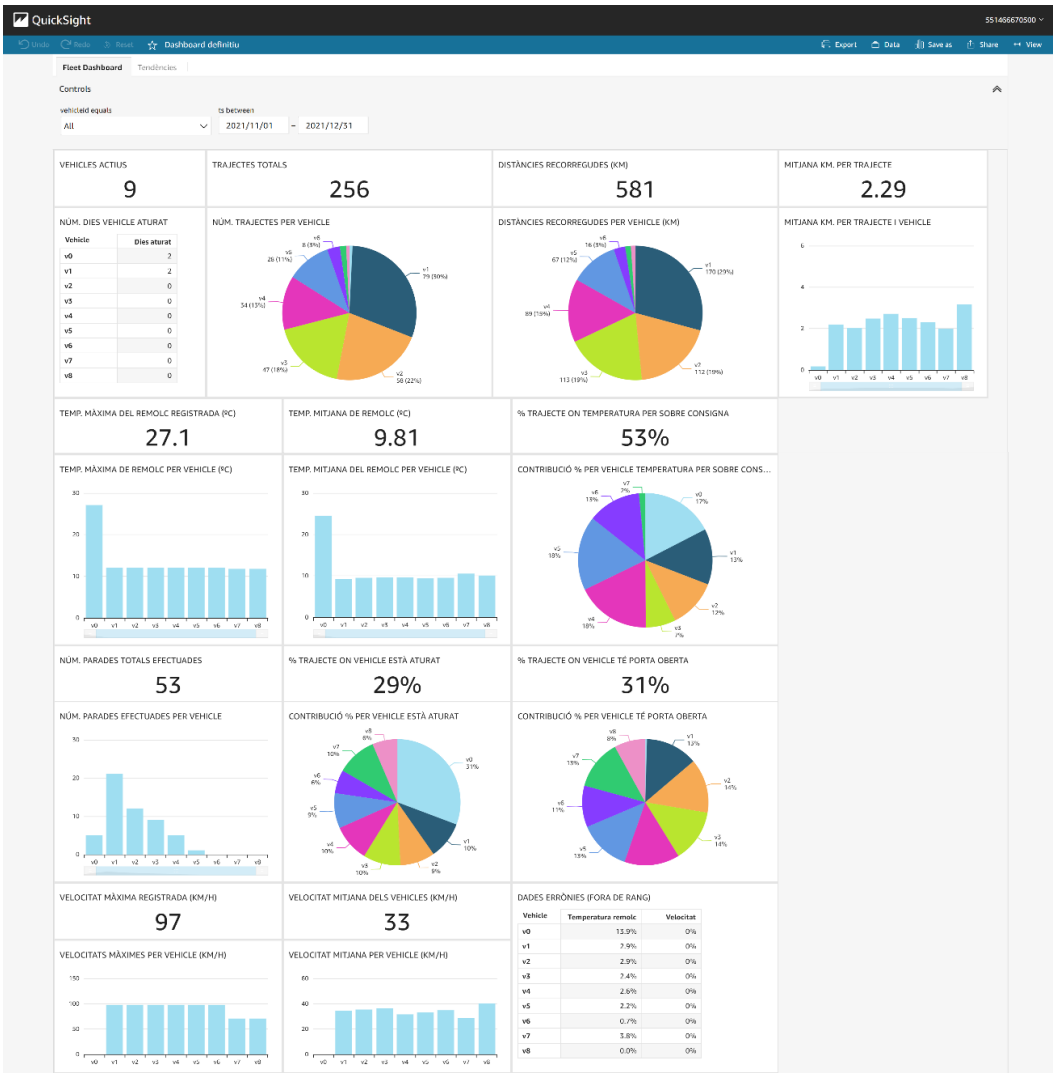


Figura 47. Dashboard principal
Elaboració pròpia

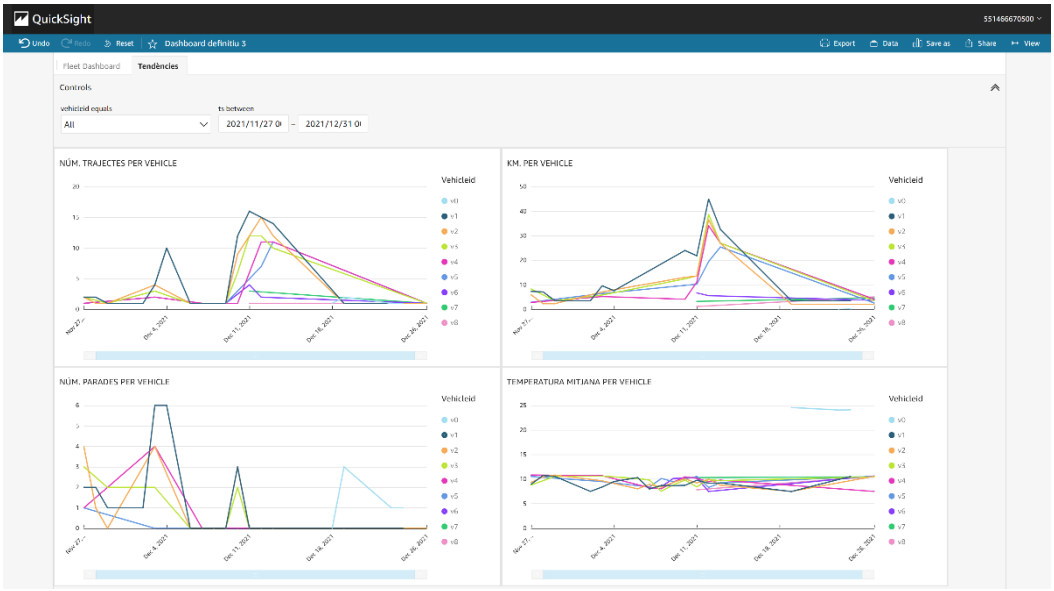


Figura 48. Dashboard de tendències
Elaboració pròpia

7. Conclusions

Amb la realització d'aquest projecte, s'ha reflectit, des d'una vessant pràctica, tot el procés necessari per crear una solució IoT destinada a la gestió de flotes pel transport frigorífic: des de que els dispositius IoT instal·lats als vehicles adquireixen les dades dels diferents sensors fins que aquestes es mostren en diversos quadres de comandament per a la gestió i presentació en temps real i per a l'anàlisi forense de la informació registrada.

S'ha corroborat que el desenvolupament d'una solució IoT d'aquestes característiques requereix coneixements tant de l'àmbit de les telecomunicacions com d'electrònica i informàtica. Ha estat necessari entendre nous conceptes com el protocol MQTT o el funcionament del xifratge TLS, aprendre a programar un microcontrolador de la família STM32 o amb llenguatge Python i comprendre el funcionament d'aplicacions com Thingsboard o dels diferents serveis que ofereix la plataforma d'AWS.

Pel que fa als objectius, s'ha aconseguit desenvolupar amb èxit les tres parts de les que constava aquest projecte. En primer lloc, s'ha implementat un prototip capaç no només d'obtenir diversos paràmetres (ubicació, velocitat, temperatura, detecció de porta oberta) i enviar-los a la plataforma d'AWS, sinó també de rebre les dades enviades per la plataforma i actuar en conseqüència sobre el vehicle.

En segon lloc, s'ha creat un simulador de vehicles que ha permès afegir a la plataforma IoT més dispositius de característiques similars al prototip. Encara que hagués estat adequat generar més dades, amb més variabilitat i més realistes, l'objectiu principal d'aquest punt ha estat comprovar la correcta ingesta i tractament de dades per part de la plataforma IoT i dels propis vehicles.

I finalment, s'han utilitzat els serveis que ofereix AWS per realitzar la tercera part del projecte, la implementació de la plataforma IoT. Per una banda, s'ha instal·lat la plataforma Thingsboad que ha permès gestionar els paràmetres i presentar la informació dels vehicles en temps real en un dashboard. I per l'altre, s'ha utilitzat Amazon QuickSight per crear un altre dashboard on analitzar la informació històrica registrada.

Thingsboard ha estat una eina molt útil per desenvolupar ràpidament tota la interfície gràfica en temps real, però té certes limitacions a l'hora d'obtenir paràmetres que no venen implementats per defecte, com conèixer el nombre de vehicles actius que hi ha en un moment determinat o obtenir els quilòmetres que realitza cada vehicle. A més, el fet que treballi amb els seus propis tòpics MQTT quan s'utilitzen els ginys de control dificulta considerablement la integració amb serveis externs com AWS IoT Core.

Pel que fa a AWS, encara que inicialment costa familiaritzar-se amb la plataforma degut a l'àmplia varietat de serveis disponibles, presenta una gran flexibilitat amb moltes opcions de configuració i una molt bona interconnexió entre els diferents serveis que ofereix.

El seguiment de la planificació s'ha pogut dur a terme de la forma prevista durant totes les PAC excepte a la tercera. En aquesta, es van haver de realitzar algunes modificacions temporals degut a la recomanació del tutor d'afegir l'anàlisi forense de dades a la plataforma IoT. Amb la realització d'aquest projecte s'ha constatat que en una plataforma IoT és important tenir constància del que està passant amb els vehicles en temps real però encara és més important poder realitzar un anàlisi posterior de les dades ja que és d'on realment es pot treure valor.

A més, també es va necessitar més temps del previst per realitzar el simulador de vehicles amb Python, ja que aquest llenguatge de programació no s'havia utilitzat fins el moment o per establir la comunicació entre el mòdul de comunicacions i el servei d'AWS IoT Core al trobar escassa documentació per la xarxa.

Per altra banda, la intenció inicial era analitzar i utilitzar una de les tecnologies LPWAN, com NB-IoT o LTE Cat-M. No obstant, després d'adquirir la targeta SIM destinada a aquestes tecnologies i d'intercanviar diversos correus electrònics amb el proveïdor no es va aconseguir obtenir connectivitat IP. No obstant, com que la targeta SIM adquirida i el mòdul de comunicacions també eren compatible amb la tecnologia GPRS, es va optar per aquesta tecnologia i així poder seguir amb el desenvolupament del projecte.

La sensorització de vehicles i la dotació de connectivitat amb el món exterior, proporcionen un ampli ventall de possibilitats a l'hora d'explorar noves línies de futur relacionades amb l'IoT.

Encara que aquest projecte s'ha centrat principalment amb la gestió i supervisió de la temperatura d'un remolc de transport frigorífic, l'addició d'altres sensors per supervisar l'estat del conductor o l'estat del vehicle podrien oferir nous paràmetres d'anàlisi. Per altra banda, la utilització de sensors sense fils també oferiria una millor flexibilitat en la seva instal·lació al no estar limitats pel cablejat de la instal·lació.

També es pot comprovar com ha estat necessari copiar uns certificats en el dispositiu IoT desenvolupat per aconseguir la comunicació amb el servei d'AWS IoT Core. No obstant, aquests certificats tenen una data de venciment que, encara que molt llunyana (any 2049 en el cas dels generats mitjançant AWS IoT), provoca que deixin d'estar operatius un cop superada. La possibilitat de que aquests dispositius es poguessin actualitzar remotament (actualització OTA o per aire) permetria, no només substituir els certificats sinó també afegir noves funcionalitats sense necessitat de connectar-los físicament.

Quant a la plataforma IoT, les possibilitats que ofereix tot el conjunt de serveis d'AWS són molt àmplies. Des d'implementar noves funcions per realitzar anàlisis de dades més complertes fins a aplicar tècniques de *Machine Learning* per obtenir prediccions i detectar, per exemple, casos de somnolència o les rutes òptimes en funció del trànsit.

Per tant, com es pot apreciar, l'IoT en el sector del transport presenta un gran potencial per explorar amb un llarg futur per endavant on la innovació abasta des del petit dispositiu instal·lat en el vehicle fins a la plataforma IoT encarregada de la gestió i el tractament de les dades.

8. Glossari

APN (*Access Point Name*): nom de la passarel·la que proporciona el servei d'accés a una xarxa de dades de comunicació sense fils la qual pot ser pública (com la dels proveïdors de telefonia mòbil) o privada.

AWS (*Amazon Web Services*): conjunt de serveis de computació i emmagatzematge en el núvol públic ofertes a través d'Internet per Amazon.com

CSV (*Comma Separated Values*): arxiu de text per representar dades en forma de taula, en què les columnes se separen per comes (o punt i coma) i les files per salts de línia.

GNSS (*Global Navigation Satellite System*): engloba qualsevol sistema de navegació per satèl·lit que proporciona posicionament geoespacial amb cobertura global, tant de forma autònoma com amb sistemes d'augmentació.

GPIO (*General Purpose Input/Output*): pin d'un xip que pot configurar-se com a entrada o sortida i utilitzar-se per una àmplia gamma de propòsits.

GPRS (*General Packet Radio Service*): mètode de transferència de dades en les xarxes de telefonia mòbil 2G. Es considera una extensió millorada del GSM.

I2C (*Inter-Integrated Circuit*): bus de comunicacions sèrie utilitzat per a comunicar microcontroladors i els seus perifèrics en sistemes integrats o comunicar circuits integrats entre sí.

IoT (*Internet of Things*): xarxa de dispositius i objectes que incorporen sensors, software i altres tecnologies amb l'objectiu de connectar i intercanviar dades amb altres dispositius i sistemes a través d'una xarxa (ja sigui privada o Internet).

JSON *(JavaScript Object Notation):* format de text senzill per emmagatzemar i intercanviar dades.

MQTT *(MQ Telemetry Transport):* protocol de missatgeria lleuger de publicació-subscripció enfocat a la connectivitat Machine to Machine (M2M).

NMEA (*National Marine Electronics Association*): protocol utilitzat pels dispositius GPS per entregar les dades i poder-se comunicar amb altres dispositius.

RPC (*Remote Procedure Call*): procés de comunicació bidireccional orientada a sol·licituds que permet enviar ordres a/des de dispositius i rebre el resultat de la seva execució.

TLS (Transport Layer Security): protocol criptogràfic, predecessor de SSL (Secure Sockets Layer), que proporciona comunicacions segures per una xarxa, normalment Internet.

USART (*Universal Synchronous and Asynchronous Receiver-Transmitter*): mòdul de comunicació sèrie altament flexible integrat en microcontroladors i plaques base, utilitzat per comunicar-se amb altres dispositius.

9. Bibliografia

[1] International Institute of Refrigeration (IIR), «The Role of Refrigeration in Worldwide nutrition (2020), 6th Informatory Note on Refrigeration and Food,» 2020. [En línia]. Available: https://iifiir.org/en/fridoc/142029. [Últim accés: 02 09 2021].

[2] Pelican BioThermal, «2019 Biopharma cold chain logistics survey,» 2019. [En línia]. Available: https://pelicanbiothermal.com/blog/2019-biopharma-cold-chain-logistics-survey. [Últim accés: 02 09 2021].

[3] DHL, «Next-Generation Wireless in Logistics Trend Report,» 2020. [En línia]. Available: https://www.dhl.com/global-en/home/insights-and-innovation/thought-leadership/trend-reports/next-generation-wireless.html. [Últim accés: 10 09 2021].

[4] D. M. Cabruja, «Sistema de seguiment de vehicles via web basat en GPS, GPRS i Arduino,» 2019. [En línia]. Available: http://openaccess.uoc.edu/webapps/o2/handle/10609/95728. [Últim accés: 07 09 2021].

[5] World Health Organization (WHO), «Temperature and humidity monitoring systems for transport operations,» 2015. [En línia]. Available: https://www.who.int/medicines/areas/quality_safety/quality_assurance/supplement_6.pdf. [Últim accés: 10 09 2021].

[6] J.C.S. dos Santos; F.O.O. Gomes; M. A. ds Santos; M. C. Felippeto de Castro; N. Grando; R. Travessini; R. S. Weirich, «Optimized ultra-low power sensor-enabled RFID data logger for pharmaceutical cold chain,» *IEEE Brasil RFID, 7-8 Oct. 2015,* 2015.

[7] Wenli Zhang; Tingting Cheng; Huamin Chen; Xiang Guo; Guohua Gao, «Design of Whole Chain Temperature Monitoring System for Raw Milk,» *2018 14th IEEE International Conference on Signal Processing (ICSP),* 2018.

[8] Fotios Zantalis, Grigorios Koulouras, Sotiris Karabetsos, Dionisis Kandris, «A Review of Machine Learning and IoT in Smart Transportation,» *Future Internet, Volume 11, Issue 4,* 2019.

[9] Vodafone España, «Barómetro IoT,» 2021. [En línia]. Available: https://www.observatorio-empresas.vodafone.es/informes/barometro-iot/. [Últim accés: 12 09 2021].

[10] Mahaveer Penna; Shivashankar; B Arjun; K R Goutham; Lohith N Madhaw; Kumar G Sanjay, «Smart fleet monitoring system using Internet of Things (IoT),» de *2017 2nd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT)*, 2017.

[11] Mariana Falco; Ignacio Núñez; Federico Tanzi, «Improving the Fleet Monitoring Management, through a Software Platform with IoT,» de *2019 IEEE International Conference on Internet of Things and Intelligence System (IoTaIS)*, 2019.

[12] Bartosz Jachimczyk, Damian Dziak, Jacek Czapla, Pawel Damps, Wlodek J. Kulesza, «IoT On-Board System for Driving Style Assessment,» *Wireless Sensors Networks in Activity Detection and Context Awareness,* 2018.

[13] A.B. Nkoro; Y.A. Vershinin, «Current and Future Trends in Applications of Intelligent Transport,» *17th International IEEE Conference on Intelligent Transportation Systems (ITSC),* 2014.

[14] Pai H. Chou; Cheng-Ting Lee; Zan-Ya Peng; Jo-Ping Li; Tong Kun Lai; Chun-Min Chang; Cheng-Hsun Yang; Yi-Lin Chen; Chin-Chung Nien; Li-Huei Chen, «A Bluetooth-Smart Insulating Container for Cold-Chain Logistics,» *2013 IEEE 6th International Conference on Service-Oriented Computing and Applications,* 2013.

[15] Eiman Mohyeldin, «Minimum Technical Performance Requirements for IMT-2020 radio interface(s),» 2017. [En línia]. Available: https://www.itu.int/dms_pub/itu-r/opb/rep/R-REP-M.2410-2017-PDF-E.pdf. [Últim accés: 08 09 2021].

[16] Marco Centenaro; Cristina E. Costa; Fabrizio Granelli; Claudio Sacchi; Lorenzo Vangelista, «A Survey on Technologies, Standards and Open Challenges in Satellite IoT,» *IEEE Communications Surveys & Tutorials,* vol. 23, núm. 3, pp. 1693 - 1720, 2021.

[17] Jasenka Dizdarević; Francisco Carpio; Admela Jukan; Xavi Masip-Bruin, «A Survey of Communication Protocols for Internet of Things and Related Challenges of Fog and Cloud Computing Integration,» *ACM Computing Surveys 51(6),* vol. 51, núm. 6, 2018.

[18] Daniel Silva, Liliana I. Carvalho, José Soares and Rute C. Sofia, «A Performance Analysis of Internet of Things Networking Protocols: Evaluating MQTT, CoAP, OPC UA,» *Appl. Sci.,* vol. 11, núm. 11, 2021.

[19] Maria K Tom, «MQTT-SN Protocol - A Review,» *International Journal of Research in Engineering, Science and Management,* vol. 1, núm. 10, 2018.

[20] Sirajuddin AhmedS. M. AbbasHina Zia, «Smart Cities—Opportunities and Challenges,» *Select Proceedings of ICSC 2019,* pp. 231-244, 2020.

[21] Controllerstech, «DS18B20 and STM32,» [En línia]. Available: https://controllerstech.com/ds18b20-and-stm32/. [Últim accés: 10 10 2021].

[22] Maxim Integrated, «DS18B20,» [En línia]. Available: https://datasheets.maximintegrated.com/en/ds/DS18B20.pdf. [Últim accés: 11 10 2021].

[23] Waveshare, «SIM7000E NB-IoT / Cat-M / EDGE / GPRS HAT,» [En línia]. Available: https://www.waveshare.com/product/iot-communication/long-range-wireless/nb-iot-lora/sim7000e-nb-iot-hat.htm. [Últim accés: 01 10 2021].

[24] IoT Cards, «SIM M2M de Telefónica,» [En línia]. Available: https://landing.iot.cards/es/es/sims-m2m-de-telef%C3%B3nica. [Últim accés: 28 09 2021].

[25] oakkar7, «AT Commands Tester,» [En línia]. Available: https://github.com/oakkar7/ATester. [Últim accés: 14 10 2021].

[26] STMicroelectronics, «STM32CubeIDE,» [En línia]. Available: https://www.st.com/en/development-tools/stm32cubeide.html. [Últim accés: 29 09 2021].

[27] SIMCom, «SIM7000 Series_AT Command Manual_V1.04,» [En línia]. Available: https://simcom.ee/documents/SIM7000x/SIM7000%20Series_AT%20Command%20Manual_V1.04.pdf. [Últim accés: 15 10 2021].

[28] SIMCom, «SIM7000 Series_MQTT_Application Note_V1.00,» [En línia]. Available: https://simcom.ee/documents/SIM7000x/SIM7000%20Series_MQTT_Application%20Note_V1.00.pdf. [Últim accés: 16 10 2021].

[29] SIMCom, «SIM7000 Series_SSL_Application Note_V1.01,» [En línia]. Available: https://www.waveshare.com/w/upload/a/a8/SIM7000_Series_SSL_Application_Note_V1.00.pdf. [Últim accés: 25 10 2021].

A. ANNEXES

A.1 Configuració d'AWS IoT Core

AWS IoT Core és el servei d'AWS IoT que actua com a *broker* en la comunicació MQTT, transmetent missatges de manera segura cap i des d'aplicacions i dispositius. A més, ofereix autenticació mútua i xifrat en tots els punts de connexió per a que les dades mai s'intercanviïn entre dispositius i AWS IoT Core sense una identitat verificada. Això permet implementar i administrar certificats i polítiques per escollir els permisos i accessos d'un dispositiu en concret o d'un grup de dispositius.

Per altra banda, AWS IoT Core també disposa d'un motor de regles que permet avaluar els missatge d'entrada publicats, transformar-los i entregar-los a un altre dispositiu o servei en el núvol. Aquestes regles es poden aplicar a dades d'un o de varis dispositius i poden prendre una o diverses accions en paral·lel.

Per utilitzar aquest servei, en primer lloc, cal registrar-se a AWS si no es disposa d'un compte. A continuació, per registrar un nou objecte IoT, cal obrir el servei AWS IoT Core i des del menú de l'esquerre *Manage>Things*, clicar sobre *Create things*, com es mostra a la Figura 49. A AWS, un *IoT Thing* és la representació del dispositiu físic al núvol i és necessari per poder interactuar amb AWS IoT. Amb aquest procés s'obté el certificat i la política necessaris per permetre que el dispositiu es connecti a AWS IoT.

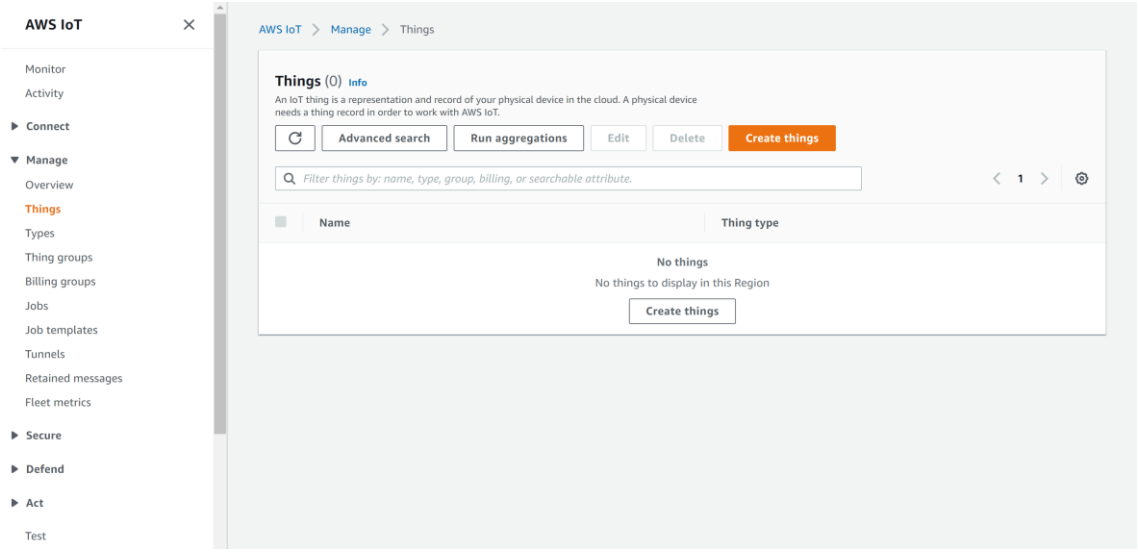


Figura 49. Creació d'un nou IoT Thing
Elaboració pròpia

Després d'escollir el nom de l'*IoT Thing*, a l'aparat de *Attach policies to certificate* cal crear una política per definir el conjunt d'accions autoritzades per als dispositius creats a AWS IoT Core. Per fer-ho, cal clicar a *Create policy*, on es designa el nom de la política i s'afegeixen les declaracions de política que defineixen els tipus d'accions que pot dur a terme un recurs determinat, ja sigui un client o tòpic.

En aquest cas, com s'observa a la Figura 50, es permet qualsevol tipus d'acció ja sigui publicar a un tòpic, subscriure-s'hi o connectar-se amb un client des de qualsevol dispositiu que tingui assignada aquesta política a través del seu certificat.

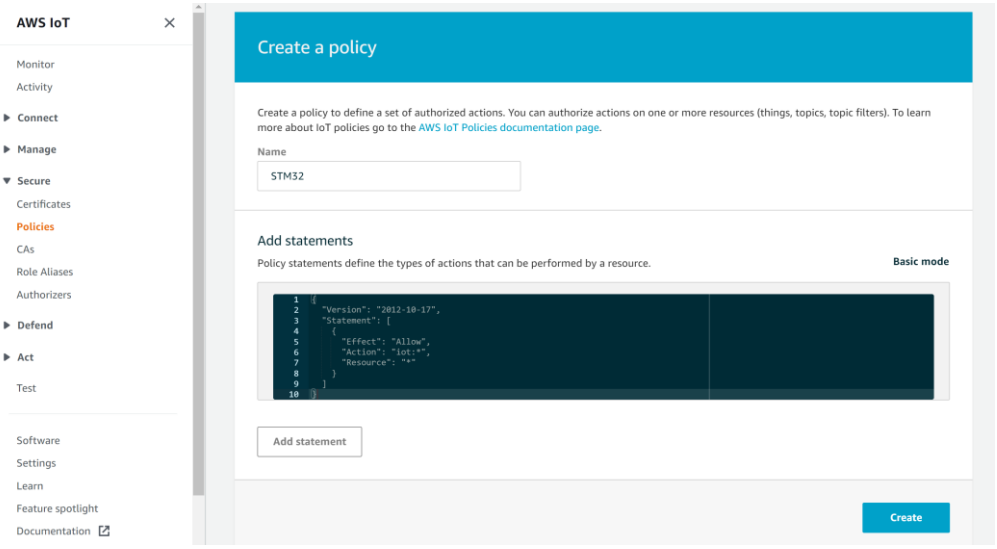


Figura 50. Creació de la política
Elaboració pròpia

Per tant, un cop creada la política, s'associa amb el dispositiu creat i s'accedeix a la pantalla de descàrrega de certificats i claus, mostrada a la Figura 51. D'aquí cal descarregar-se el certificat del dispositiu (*Device certificate*) i la clau privada (*Private key file*). Com a certificat Root CA, per a que funcioni amb el dispositiu utilitzat en el prototip, cal descarregar-se el certificat d'autenticació del servidor de AWS IoT Core anomenat *Certificado de CA raíz G5 principal y público de clase 3 de VeriSign*.

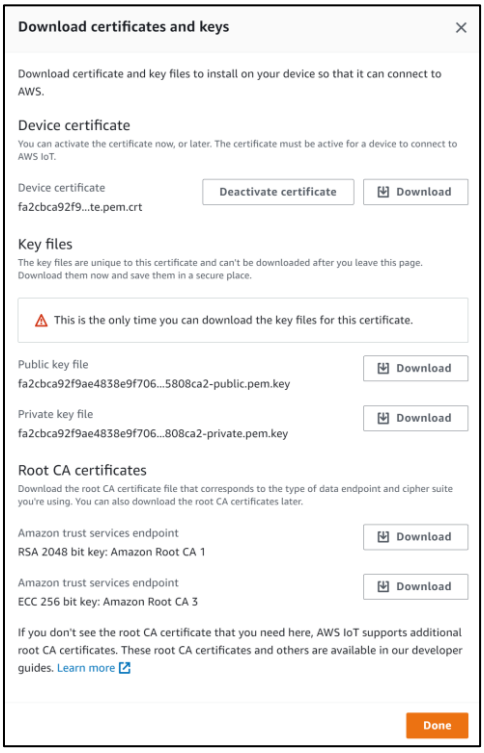


Figura 51. Descàrrega de claus i certificats
Elaboració pròpia

Això és necessari perquè el broker d'AWS IoT xifra totes les comunicacions mitjançant el protocol TLS versió 1.2, que és el successor de SSL (*Secure Sockets Layer*). El protocol TLS és utilitzat per garantir la confidencialitat dels protocols d'aplicació com MQTT o HTTP.

En el cas d'MQTT, TLS xifra la connexió entre el dispositiu i el broker i AWS IoT utilitza l'autenticació de client TLS per identificar els dispositius a partir d'un certificat X.509. Això permet protegir les dades en trànsit (a mesura que viatgen cap a i des de l'AWS IoT) o mentre s'emmagatzemen en dispositius o altres serveis d'AWS i verificar que realment la comunicació s'estableix amb AWS IoT Core.

Així doncs, un cop es disposa d'aquests tres arxius, cal copiar-los dins el mòdul de comunicacions SIM7000E, concretament, dins la carpeta anomenada *customer* mitjançant un programa disponible per Windows anomenat *QPST tool*. Per fer-ho, cal connectar el mòdul per USB a un ordinador, executar el programa *QPST tool* i anar al menú *Start Clients > EFS Explorer* i clicar l'opció del menú superior *Alternate File System* (símbol d'un disc de color verd), tal i com es mostra a la Figura 52.

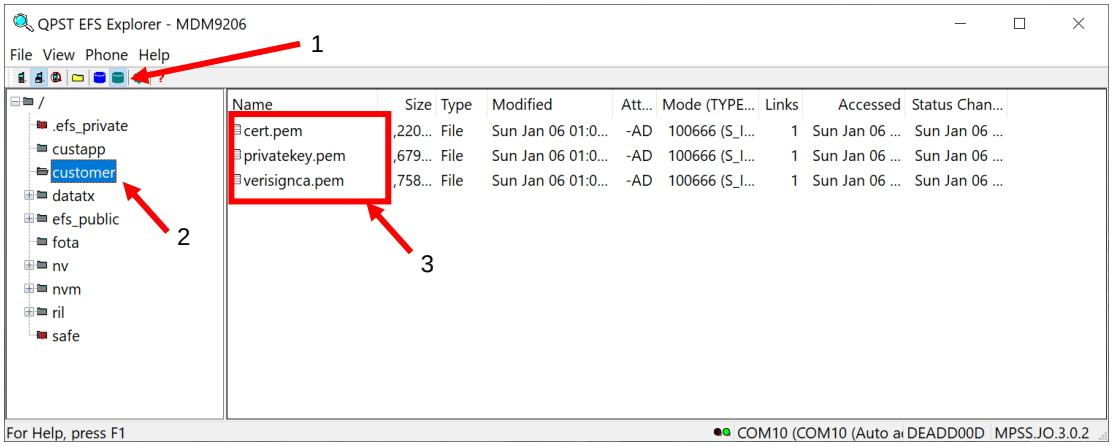


Figura 52. *QPST tool*
Elaboració pròpia

Abans de copiar els fitxers al mòdul i agafant com a referencia els noms que apareixen a la Figura 51, serà necessari renombrar aquests fitxers descarregats com es mostra a la Taula 15, ja que si els noms dels arxius són molt llargs poden donar problemes.

	Nom orginal	Nom donat
Device certificate	fa2cb...te.pem.crt	cert.crt
Private key file	...808ca2-private.pem.key	private.key
Root CA	...Certification-Authority-G5.pem	rootCA.pem

Taula 15. Renombrament de certificats i claus

Per comprovar que els missatges MQTT arriben correctament al compte d'AWS, el servei d'AWS IoT disposa d'un client de prova MQTT des del qual pot subscriure's al tòpic en qüestió. Per accedir-hi, des del menú de l'esquerre d'aquest servei, cal anar a *Test>MQTT test client*, clicar sobre *Subscribe to a topic* i afegir el filtre de tòpic. A continuació, al clicar el botó *Subscribe*, s'hauria de veure com arriben els missatges a mesura que es van enviant des del dispositiu.

A.2 Configuració d'AWS EC2 i Lambda

Els passos a seguir per a la creació de la instància EC2 i la instal·lació de Thingsboard estan perfectament detallats a https://thingsboard.io/docs/user-guide/install/pe/aws-marketplace.

A l'hora d'escollir el tipus d'instància i per tal que l'aplicació vagi suficientment fluida, es recomana utilitzar com a mínim una instància del tipus *t2.medium*, la qual disposa de 2 nuclis virtuals i 4 GB de memòria. També cal tenir en compte que la regió de la instància ha de ser la mateixa en què s'utilitza *AWS IoT Core*. Un cop hagi finalitzat la preparació de la instància, es poden comprovar aquests paràmetres accedint al servei d'AWS EC2, tal com s'observa a la Figura 53.

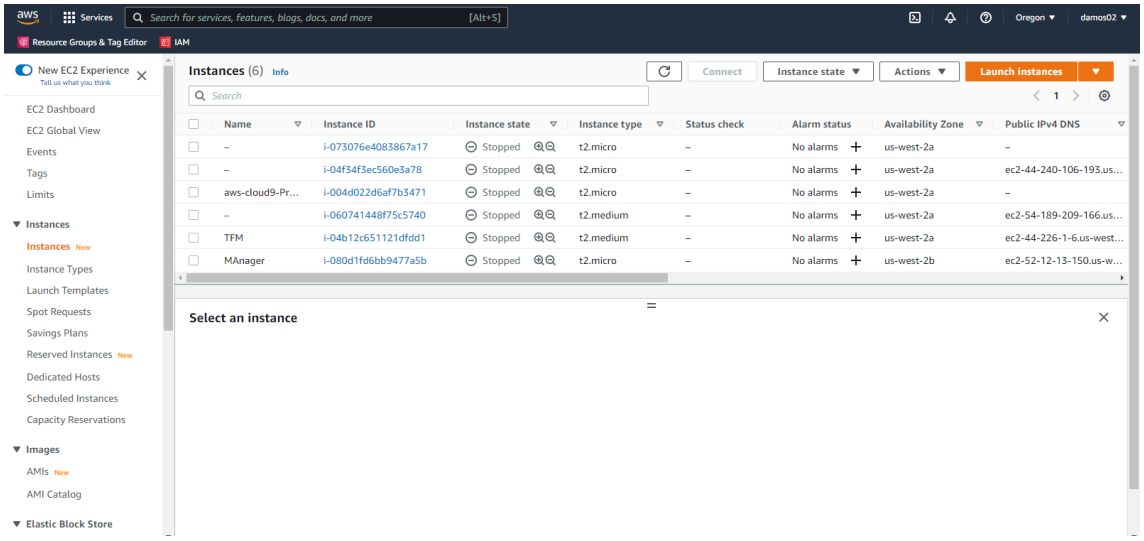


Figura 53. Instàncies d'AWS EC2
Elaboració pròpia

Per altra banda, a l'hora d'adquirir una llicència Thingsboard s'ha optat per la subscripció *Thingsboard PE Maker* que costa 10$ al mes i admet fins a 10 dispositius, suficient per provar la solució IoT desenvolupada en aquest projecte. Cal recordar que, una vegada realitzat el pagament, es proporciona un codi que cal copiar en el fitxer de la ubicació indicada a l'enllaç anterior. Per fer-ho, cal accedir a la instància a través de SSH. Un cop dins la màquina virtual, l'usuari per defecte per entrar a les màquines Linux d'AWS EC2 es *ubuntu*.

Com s'ha comentat, el cost d'aquestes màquines virtuals depèn del tipus d'instància creada i del nombre d'hores d'ús, de manera que com més potent sigui la màquina, més car serà el cost per hora d'ús. Per tant, per minimitzar el cost i per disposar d'una aplicació totalment integrada i transparent per l'usuari sense la necessitat d'haver d'accedir a la plataforma d'AWS cada vegada que es vulgui engegar o aturar la màquina virtual, s'ha considerat oportú iniciar-la de forma automatitzada des del propi simulador de vehicles creat.

Mitjançant aquest procediment també es passa el nombre de vehicles que es volen simular com a paràmetre al RPC Manager, valor que ha escollit l'usuari en el simulador de vehicles.

A la Figura 54 es mostra el diagrama de blocs del procés seguit per iniciar la màquina virtual i executar el programa RPCManager.

Figura 54. Procés d'inicialització de la instància EC2 i RPCManager
Elaboració pròpia

La manera que s'ha plantejat la inicialització de la instància EC2 que conté la plataforma IoT, aprofitant la utilització del protocol MQTT, ha estat a través de la publicació d'un missatge MQTT a un tòpic determinat al broker d'AWS IoT Core.

Així doncs, quan a la interfície del simulador es prem el botó START, es publica al tòpic *tfm/commands/start/values* un missatge MQTT que conté el nombre de vehicles a simular que ha escollit l'usuari des de la mateixa interfície. A *AWS IoT Core* es crea una regla, que se l'ha anomenat *StartMachine*, de manera que quan es rebin missatges en aquest tòpic, s'enviïn a una funció Lambda, que en aquest cas, s'ha anomenat *StartInstance*, tal i com s'observa a la Figura 55.

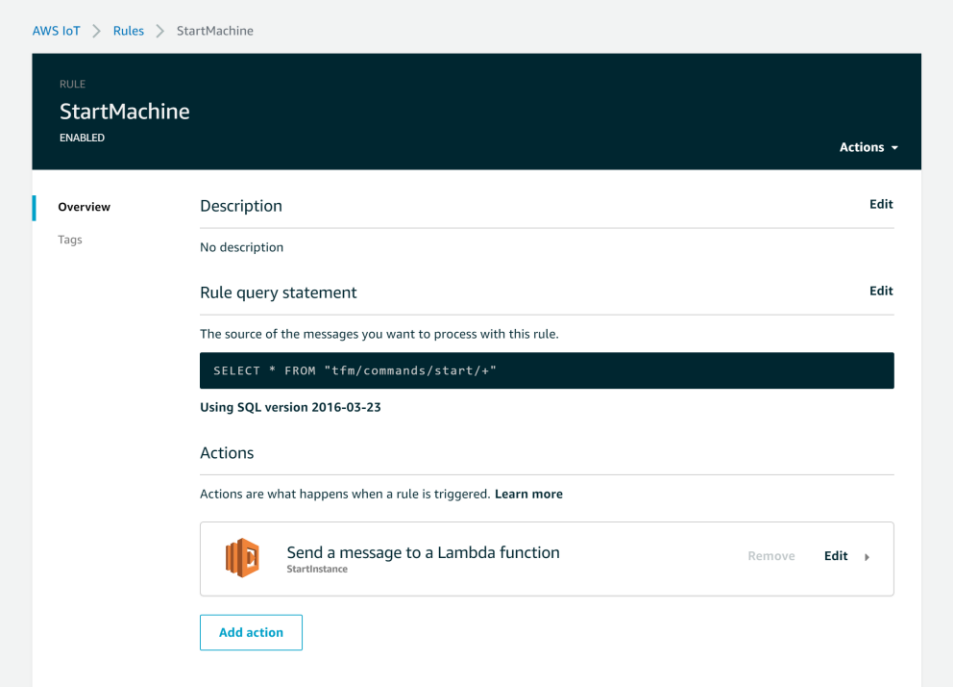


Figura 55. Regla per inicialitzar la instància EC2

Lambda és un altre servei d'AWS sense servidor i basat en esdeveniments que permet executar codi per pràcticament qualsevol tipus d'aplicació o servei backend sense necessitat d'aprovisionar o administrar servidors. A més, combinat amb la llibreria Boto3, permet interaccionar amb altres serveis d'AWS com en aquest cas, que s'ha utilitzat per iniciar o aturar una instància EC2 d'AWS.

En aquest cas, a la funció *StartInstance* s'hi ha definit un disparador (*trigger*) que s'activa quan rebi el missatge d'AWS IoT anteriorment definit. La pròpia funció creada farà iniciar la instància que conté la plataforma IoT i a la vegada passarà el paràmetre del nombre

de vehicles a la funció del RPCManager, allotjada també dins la mateixa instància, i l'executarà. Aquesta romandrà en funcionament fins que no rebi l'ordre d'aturada.

Quan es prem el botó STOP des de la interfície del simulador, s'actua d'una manera similar com es pot veure en el diagrama de blocs de la Figura 56.

Figura 56. Procés d'aturada de la instància EC2 i RPCManager
Elaboració pròpia

De manera similar, quan a la interfície del simulador es prem el botó STOP, es publica al tòpic *tfm/commands/stop/values* un missatge MQTT sense contingut. La publicació d'aquest tòpic simplement serveix per indicar que es vol aturar la instància. Així doncs, a *AWS IoT Core* es crea una altra regla, aquest cop anomenada *StopMachine*, encarregada d'enviar el missatge a la funció Lambda.

Lambda quan detecti l'arribada d'aquest missatge gràcies al disparador configurat, executarà la funció anomenada *StopInstance*, mostrada a la Figura 57.

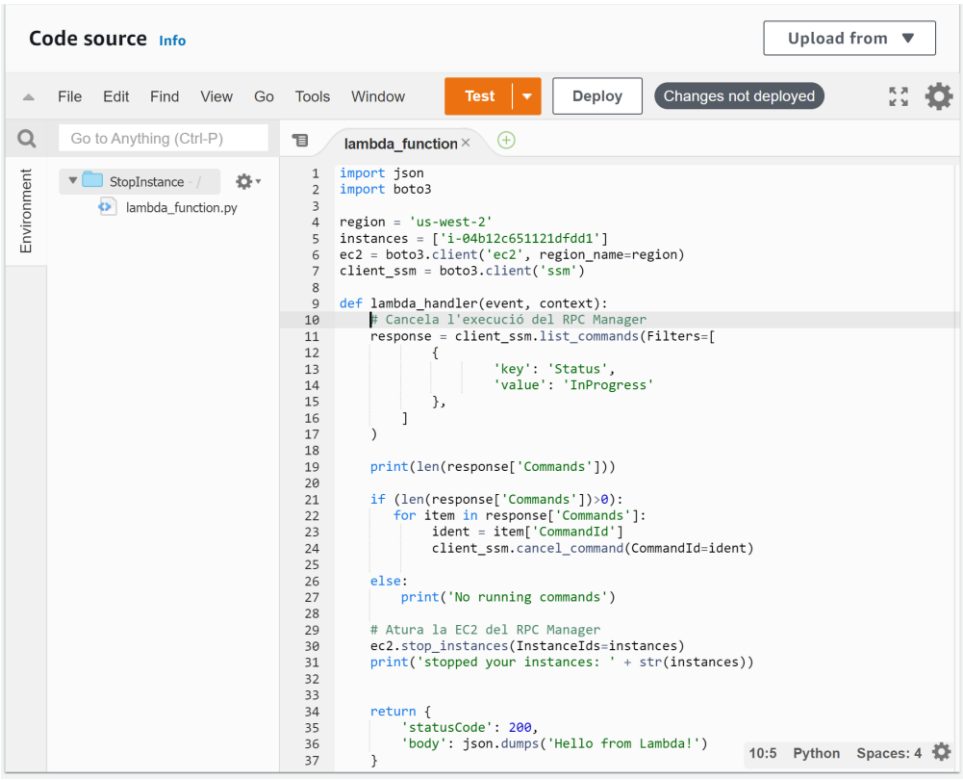


Figura 57. Codi per aturar la instància EC2
Elaboració pròpia

Com es pot veure a les línies de codi anteriors, el que es fa en primer lloc és comprovar si el programa RPCManager s'està executant i si és el cas, s'atura. A continuació, s'atura la instància EC2.

És important que cada vegada que es finalitza una simulació, abans de tanca la interfície gràfica, s'accioni el botó de STOP ja que d'aquesta manera s'atura la instància i el codi que s'estava executant en aquesta i AWS deixa de computar hores d'ús.

A.3 Comandes AT utilitzades

Les comandes AT que s'han utilitzat en aquest projecte relacionades amb la connectivitat del mòdul de comunicacions són les que es mostren a la Taula 16.

AT		
Funció	Verifica la comunicació amb el mòdul a través del port sèrie	
Paràmetres	-	
Resposta	OK o ERROR	
AT+CFUN=1		
Funció	Estableix l'estat del mòdul de comunicacions	
Paràmetres	L'1 indica funcionalitat complerta (per defecte)	
Resposta	OK o +CME ERROR	
AT+CNMP=<mode>		
Funció	Selecciona la xarxa preferida de comunicacions	
Paràmetres	<mode>	2 Automàtic 13 Només GSM 38 Només LTE
Resposta	OK o +CME ERROR	
AT+CMNB=<mode>		
Funció	Selecciona la xarxa preferida entre CAT-M i NB-IoT	
Paràmetres	<mode>	1 CAT-M 2 NB-IoT 3 CAT-M i NB-IoT
Resposta	OK o +CME ERROR	
AT+CREG?		
Funció	Comprova si s'ha registrat a la xarxa	
Paràmetres	-	
Resposta	+CREG: 0,1 (indica que s'ha registrat correctament)	
AT+CSQ		
Funció	Qualitat de la senyal	
Paràmetres	-	
Resposta	+CSQ: <value>	2 a 9: Molt dolenta 10 a 14: Dolenta 15 a 19: Bona 20 a 30: Molt bona
AT+CNACT=1,"internet.easym2m.eu"		
Funció	Activa la connectivitat IP	
Paràmetres	L'1 indica que es vol activar *"internet.easym2m.eu"* és l'APN proporcionat per l'operador	
Resposta	OK +APP PDP: ACTIVE o +APP PDP: DEACTIVE	

Taula 16. Comandes AT per configurar el mòdul de comunicacions

Les comandes AT que s'han utilitzat en aquest projecte relacionades amb la connectivitat GPS es mostren a la Taula 17.

AT+CGNSPWR=1	
Funció	Habilita el GPS
Paràmetres	L'1 indica habilitació dle GPS
Resposta	OK o ERROR
AT+CGNSINF	
Funció	Obté la trama NMEA del GPS
Paràmetres	-
Resposta	+CGNSINF: Trama NMEA (detallada a la Taula 9)

Taula 17. Comandes AT per configurar el mòdul GPS

Les comandes AT que s'han utilitzat en aquest projecte relacionades amb el protocol MQTT i SSL es mostren a la Taula 18.

AT+CSSLCFG="convert",2,verisignca.pem		
Funció	Converteix el *verisignca.pem* i el guarda al sistema d'arxius	
Paràmetres	El 2 indica que s'ha de convertir un certificat del tipus root CA *verisignca.pem* és el certificat root CA	
Resposta	OK o ERROR	
AT+CSSLCFG="convert",1,cert.pem,privatekey.pem		
Funció	Converteix *cert.pem* i *privatekey.pem* i el guarda al sistema d'arxius	
Paràmetres	L'1 indica que s'ha de convertir el certificat del dispositiu. *cert.pem* és el certificat del dispositiu *privatekey.pem* és la clau privada de *cert.pem*.	
Resposta	OK o ERROR	
AT+CSSLCFG="sslversion",0,3		
Funció	Estableix la versió a utilitzar del protocol SSL	
Paràmetres	El 0 indica el fitxer de configuració El 3 estableix la versió TLS1.2	
Resposta	OK o ERROR	
AT+SMSSL=1,verisignca.pem,cert.pem		
Funció	Estableix els certificats a utilitzar	
Paràmetres	L'1 indica el perfil on es guarda la configuració *verisignca.pem* és el certificat root CA *cert.pem* és el certificat del dispositiu	
Resposta	OK o ERROR	
AT+SMCONF="URL","xxxxxx75uhfjgw.iot.us-west-2.amazonaws.com","8883"		
Funció	Estableix la URL i el port del servidor MQTT El port per defecte és 8883 per MQTT amb SSL i 1883 sense SSL.	
Resposta	OK o ERROR	
AT+SMCONF="clientid","STM32"		
Funció	Estableix la id de connexió del client definit a AWS IoT	
Resposta	OK o ERROR	
AT+SMSTATE?		
Funció	Consulta l'estat de la connexió MQTT	
Resposta	+SMSTATE: <status> OK	<status> 0 Desconnectat 1 Connectat
AT+SMCONN		
Funció	Estableix la connexió MQTT un cop està tot configurat	
Paràmetres	OK o ERROR	

Taula 18. Comandes AT per configurar el protocol MQTT

Les comandes AT relaciones amb el protocol MQTT que s'han utilitzat en aquest projecte per a la publicació i subscripció de dades a un tòpic es mostren a la Taula 19.

AT+SMSUB="tfm/vehicles/v0/setpoint",0	
Funció	Es subscriu al topic definit per recepcionar el paquet MQTT
Paràmetres	*"tfm/vehicles/v0/setpoint"* és el tòpic referent a la consigna de temperatura 0 indica el nivell de QoS del paquet
Resposta	OK o ERROR +SMSUB: "tfm/vehicles/v0/setpoint","73"
AT+SMSUB="tfm/vehicles/v0/fans",0	
Funció	Es subscriu al topic definit per recepcionar el paquet MQTT
Paràmetres	*"tfm/vehicles/v0/fans"* és el tòpic referent l'estat de l'equip de refrigerador 0 indica el nivell de QoS del paquet
Resposta	OK o ERROR +SMSUB: "tfm/vehicles/v0/fans","1"
AT+SMPUB="tfm/vehicles",21,1,1	

> {"temp_int":99.9, "temp_ext":99.9, "latitude":41.123456...}	
Funció	Envia paquet MQTT al topic definit
Paràmetres	*"tfm/vehicles"* és el tòpic al qual va dirigit el paquet *21* és la longitud de la cadena de caràcters a enviar 1 indica el nivell de QoS del paquet 1 indica retenció de paquets activat
Resposta	OK o ERROR

Taula 19. Comandes AT de publicació/subscripció a tòpics MQTT