Treball Final de Grau

Implementació d'un sistema domòtic amb assistent de veu que protegeix la privacitat

Grau d'Enginyeria Informàtica

-Plataforma GNU/Linux-

Autor: Daniel Amiguet Romero

Agraïments

A tots els professors que m'han guiat per recórrer aquest camí i a la UOC pel seu model d'ensenyament i flexibilitat, sense el qual mai hagués estat possible estudiar aquest grau.

A la meva família, per la confiança sense límits amb mi.

I sobretot a Sandra, perquè el seu suport, paciència i ànims han estat cabdals.

Resumen

Este estudio realizará un análisis de las diferentes alternativas para implementar un sistema domótico con GNU / Linux. Nos centraremos en sistemas de software libre, es decir, el software el cual tenemos acceso al código fuente, porque queremos mantener protegida la información personal.

Pondremos en funcionamiento y configuraremos un sistema domótico con asistente de voz, donde integraremos varios dispositivos, para conseguir un sistema totalmente funcional, sin perder el control de nuestros datos personales.

Palabras clave: GNU/Linux, código abierto, domótica, asistente de voz, protección de datos

Abstract

This study will perform an analysis of the different alternatives to implement a home automation system with GNU/Linux. We will focus on free software systems, in other words, software which we have access to the source code, because we want to keep personal information protected.

We will put into operation and configure a home automation system with a voice assistant, where we will integrate several devices, to achieve a fully functional system, without losing control of our personal data.

Keywords: GNU/Linux, open source, home automation, voice assistant, data protect

Índex

Índex de taules

Índex de figures

1 Introducció

1.1 Descripció general

En aquest Treball Final de Grau, TFG, es realitzarà una anàlisi de les diferents alternatives per implementar un sistema domòtic amb GNU/Linux. Ens centrarem en sistemes de programari lliure, és a dir, programari el qual tenim accés al codi font, perquè es desitja mantenir protegida la informació personal.

Posarem en funcionament i configurarem un sistema domòtic amb un assistent de veu, on integrarem diversos dispositius, per aconseguir un sistema totalment funcional, sense perdre el control de les nostres dades personals.

1.2 Justificació del TFG

En el darrers anys, la implementació de sistemes d'automatització als nostres habitatges s'ha incrementat considerablement, sobretot, arrel de l'adopció d'altaveus intel·ligents com *Amazon Echo* o *Google Hom*e. Aquests dispositius ens faciliten la domotització de les nostres llars, gràcies a la senzillesa de configuració i gestió de la resta de dispositius connectats, com ara llums, càmeres, sensors, etc.

Desgraciadament, aquests altaveus es troben connectats contínuament i envien dades a servidors externs, per donar resposta a les nostres comandes. Aquest fet posa en risc la privadesa i la protecció de les dades personals, tot i que aquestes empreses posen l'accent en oferir major control de les dades que s'usen als usuaris.

Atesa la importància de les dades personals, i la capacitat de controlar l'ús que es fan d'aquestes, l'única opció del tot segura per a nosaltres, és adoptar sistemes programari lliure a les nostres cases. És primordial tenir control de la informació, sense menysprear la capacitat de modificar el codi per adaptar-lo a les nostres necessitats.

1.3 Objectius

El objectiu d'aquest TFG és analitzar les distintes opcions disponibles per implementar un sistema domòtic, amb sistemes GNU/Linux i programari lliure. Posarem èmfasis en les tecnologies de reconeixem de veu o assistents personals, i de control domèstic, que ens permetin protegir la nostra privadesa.

Estudiarem les tecnologies més utilitzades de comunicació i el maquinari disponible per a la nostra implementació. Examinarem i seleccionarem un programari de reconeixement de veu i un programari de control domèstic.

Iniciarem la instal·lació d'aquest binomi en un entorn domèstic i implementarem unes tasques d'automatització per, *a posteriori*, comprovar-ne el funcionament. Al finalitzar, farem una valoració del estat actual d'aquest sistema i la facilitat d'instal·lació i d'ús.

1.4 Enfocament i metodologia

Donada la importància de les dades personals, la implementació del sistema serà, per força, baix el paraigua del programari lliure. Per aconseguir-ho, emprarem un maquinari de baix cost i dimensions. Es farà una anàlisi de les diferents alternatives disponibles, que en els darrers anys s'ha popularitzat molt per a projectes casolans.

En segon lloc, analitzarem els protocols de comunicació disponibles per als sistemes domòtics. Estudiarem les alternatives de programari per al control domèstic, quines possibilitats tenen, avantatges i inconvenients de cadascun d'ells. També farem un estudi en profunditat de les diferents tecnologies de reconeixem de veu.

Per al triatge de les tecnologies es tindran en compte l'accés al codi, els idiomes suportats, la integració amb la resta de dispositius i, sobretot, tenir control en tot moment de les dades personals. Per complir amb la protecció de les dades, prioritzaren sistemes que no necessiten connexió a internet.

Una volta seleccionat, instal·larem el programari necessari i configurarem els dispositius externs. Tant el programari com els dispositius externs s'han de

poder comunicar a traves d'algun protocol de comunicació. Per obtenir un resultat satisfactori, cal aconseguir unes funcions bàsiques similars a altres productes més comercials.

1.5 Planificació i temporització

NOM	DURACIÓ	INICI	FI
PAC1: Pla de treball	14 dies	21/09/20	08/10/20
Elecció del Projecte	5 dies	21/09/20	25/09/20
Objectius	2 dies	28/09/20	29/09/20
Metodologia	4 dies	30/09/20	05/10/20
Planificació	2 dies	06/10/20	07/10/20
Redacció document PAC1	1 dies	08/10/20	08/10/20
PAC2: DISSENY I IMPLEMENTACIÓ	17 dies	09/10/20	02/11/20
Estudi maquinari i controladors	3 dies	09/10/20	13/10/20
Estudi Protocols de comunicació	3 dia	14/10/20	16/10/20
Estudi Control Domèstic	5 dies	19/10/20	23/10/20
Capítols Introductoris	3 dies	26/10/20	28/10/20
Redacció document PAC2 i memòria	3 dies	29/10/20	02/11/20
PAC3: RESULTATS I ANÀLISIS	20 dies	03/11/20	30/11/20
Estudi i Adquisició de dispositius	2 dies	03/11/20	04/11/20
Estudi assistents de veu	5 dies	05/11/20	11/11/20
Elecció de sistema i motivacions	7 dies	12/11/20	20/11/20
Instal·lació del sistema part 1	1 dia	23/11/20	23/11/10
Redacció document PAC3 i memòria	5 dies	24/11/20	30/11/20
PAC4: LLIURAMENT FINAL DE LA MEMÒRIA	15 dies	01/12/20	21/12/20
Instal·lació del sistema part 2	1 dia	01/12/20	01/12/20
Configuració del sistema	2 dies	02/12/20	03/12/20
Automatitzacions	4 dies	04/12/20	09/12/20
Redacció memòria	8 dies	10/12/20	21/12/20

1.6 Diagrama de Gantt

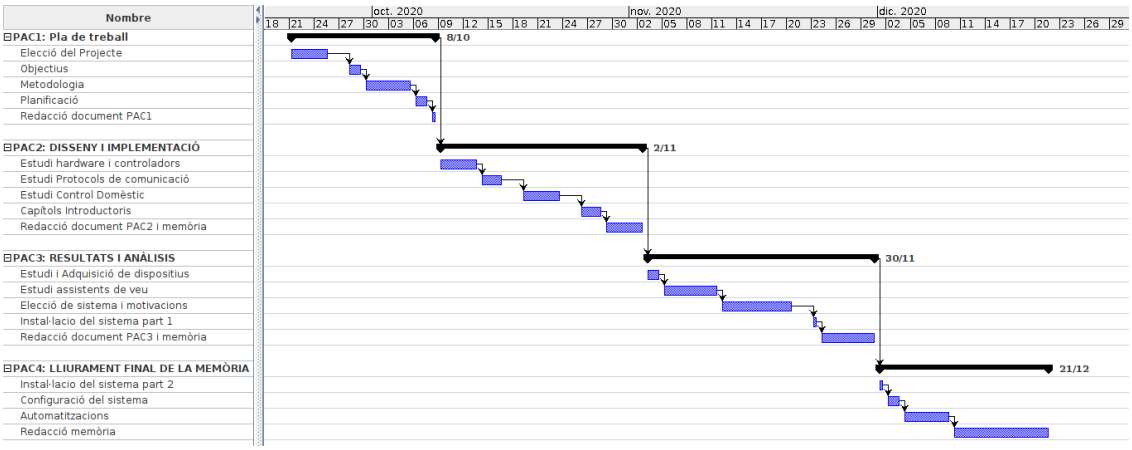

Figura 1. Diagrama de Gantt

2 GNU/Linux i la privacitat

2.1 Què és GNU/Linux?

Un sistema GNU/Linux és la combinació de dos grans projectes, GNU i Linux. Generalment, aquests dos termes se solen confondre o es denomina a tot el conjunt com a Linux. Sigui com vulgui, el camí d'aquests ha estat unit al llarg del temps i aquesta unió ha enfortit ambdós projectes irremeiablement.

Linux és un *Kernel* que funciona com Unix. El *kernel* o nucli és la part fonamental de qualsevol sistema operatiu. Amb independència del sistema que elegisques, tots empren un nucli, que és el que s'encarregarà de la comunicació entre el programari i el maquinari de qualsevol computadora. El nucli és el primer programa que s'executa just després del carregador d'arrencada. Administra la resta del inici del sistema operatiu, així com les sol·licituds de Entrada/Sortida del programari, i gestiona la memòria i perifèrics.

Linux va sorgir en 1991, pel finlandès Linus Torvalds. Aquest sistema va obtenir molta acceptació per programadors de tot el món, i una gran quantitat d'ells hi participen activament en el desenvolupament. Es tracta de programari lliure en la major part del codi, però, existeixen fragments de codi privatiu per ampliar els dispositius suportats.

Perquè el nucli funcione com un sistema operatiu complet, és necessari afegir-li un programari bàsic per donar-li funcionalitats. GNU, del acrònim *GNU's Not Unix*, és una col·lecció d'aquests programes informàtics, format completament per programari lliure. Ha estat liderat des de 1984 per Richard Stallman, gran defensor del moviment social del programari lliure, i responsable, entre altres, del compilador GCC .

En l'actualitat, GNU/Linux ha penetrat en molts àmbits diferents; des de computadores personals amb les distribucions populars d'Ubuntu o Linux Mint, fins a supercomputadors, sense oblidar-nos de sistemes encastats. És a la supercomputació on ha aconseguit una gran hegemonia, conseqüència del bon

rendiment i la capacitat d'adaptació del codi segons les necessitats. GNU/Linux està present des de 2004 en la majoria dels supercomputadors en cada rànquing publicat per TOP500, i en la totalitat des de novembre de 2018.

2.2 Seguretat al codi obert

El codi obert sempre ha tingut la reputació de ser més segur que el programari privatiu. Aquesta generalització és errònia. Un codi no és més segur per ser obert, ni és més insegur per ser codi tancat.

A qualsevol programari lliure, els usuaris tenen la llibertat d'executar, copiar, distribuir, estudiar, modificar i millorar el codi. És aquesta llibertat la que ens ofereix major seguretat. El que podem assegurar per tant, és que el programari lliure es pot auditar i corregir si es dubta de la seguretat que ofereix. Aquesta apertura del codi no implica tenir millor disseny, només que amb un codi tancat no tenim control que es fa amb les nostres dades.

No hem de menysprear el recolzament de grans corporacions al ecosistema GNU/Linux. Segons el '2020 Linux Kernel History Report', de la organització sense ànim de lucre Linux Foundation, des de 2007, 1.730 organitzacions diferents han contribuït al codi del nucli Linux. Sols les vint que han contribuït més, representen un 68% del *commits*, entre les quals es troben Intel, Red Hat, IBM, Google, Samsung o Oracle.

Per què es important aquest suport? Ens hem d'oblidar que el codi obert està escrit per aficionats i que no té un manteniment. Les empreses, com les nombrades i com a usuàries d'aquest programari, són les primeres interessades en fer-les més segures i mantenir-les al llarg del temps.

> *«If you run GNU/Linux or another free operating system, and if you avoid installing proprietary applications on it, then you are in charge of what your computer does. If a free program has a malicious feature, other developers in the community will take it out, and you can use the corrected version»*
> *Free Software, Free Society. Chapter 37: Can You Trust Your Computer?*
> *Richard M. Stallman*

2.3 Privadesa al codi obert

Per obtenir un sistema que protegisca les nostres dades, necessàriament ha de ser segur. Però, assegurar-nos que el codi n'és d'atacs externs, no significa que l'ús que es fa de les dades sigui l'apropiat.

En els darrers anys, han sorgit noves legislacions en matèria de protecció de dades, com la RGPD a nivell europeu. Malgrat tot, continuem estant a mercè d'empreses externes. Per determinar si les dades personals en un sistema informàtic són compartides amb tercers, tornem a necessitar tenir accés al codi.

Un cas de recent actualitat, arrel de la pandèmia del COVID -19 i del aplicatiu Radar COVID, és que membres de la comunitat acadèmica espanyola han realitzat un manifest a favor de la transparència en el desenvolupament de programari públic. Aquest exposa la necessitat d'implementar processos oberts, com la publicació de documentació i del codi, per identificar punts a millorar i contribuir al desenvolupament. 'No hi ha tecnologia sense errors i per tant és necessari un escrutini multidisciplinari per aconseguir el millor resultat', manifesta el document.

3 Domòtica

Els sistemes domòtics són aquells sistemes capaços d'automatitzar un habitatge o edifici amb el fi de millorar la gestió energètica i la qualitat de vida dels seus residents. Un sistema domòtic pot controlar dispositius de seguretat, d'il·luminació, climàtics, electrodomèstics o sistemes d'entreteniment.

3.1 Introducció històrica

A la dècada dels setanta van sorgir els primers dispositius d'automatització, el quals empraven el protocol de comunicació X10. Però no va ser fins a la dècada dels vuitanta quan la domòtica es va utilitzar a nivell comercial. Començaren a aparèixer el primers dispositius com detectors de moviment, controls de termòstat o portes automàtiques.

A mitjans de la dècada dels noranta i al inici del nou mil·lenni, els protocols de comunicació s'unificaren i globalitzaren. Els habitatges intel·ligents es tornaren més i més populars, i sorgiren tot un ecosistema de tecnologies, aparells i dispositius cada vegada més comuns i assequibles. Sobretot, graciés als sistemes sense fils, com ZigBee o Z-Wave, l'adopció de la domòtica pel públic general es va popularitzar.

3.2 Funcions dels sistemes domòtics

Les funcions d'un sistema domòtic poden ser molt extensos i variats.

- Gestió d'energia: la gestió dels sistemes de climatització/calefacció, escalfadors i il·luminació per mitja d'automatitzacions, pot ajudar-nos a reduir el consum elèctric.
- Automatització de tasques: activar la climatització depenent de la temperatura, gestionar les persianes segons la llum exterior, el reg automàtic, o encendre aparells a una hora determinada, pot incrementar el confort en el habitatge.

- Seguretat: control d'accessos, visualització remota de la llar, detecció de fuites d'aigua o gas, o emular la presència de gent, poden ser exemples en matèria de seguretat.
- Control remot: tenir accés a qualsevol aparell a traves de internet per consultar estats, com la temperatura, o realitzar alguna tasca, com encendre la rentadora.

3.3 Arquitectura

Per controlar el sistema es poden seguir diverses arquitectures.

- Centralitzada: el sistema centralitzat es gestiona amb un controlador central, que rep la informació dels sensors i envia les ordres als actuadors. Aquesta tipologia és molt fàcil d'implementar, encara que, en cas de fallada del controlador, el sistema deixa de funcionar per complet.
- Distribuïda: tant els sensors com els actuadors poden funcionar com a controladors, i per tant, són capaços d'analitzar la informació i executar les ordres. Amb aquesta arquitectura, els dispositius es troben connectats a través d'un bus central.
- Mixta: el sistema es divideix per zones, i cadascuna d'elles es gestiona de forma centralitzada.

3.4 Seguretat en la domòtica

Els dispositius domòtics presenten diversos riscos de seguretat, que són comuns a la resta de xarxes de computadors. Per les característiques de la domòtica, el accés físic als dispositius o les comunicacions amb xarxes sense fils, són dos dels aspectes més importants.

Qualsevol xarxa a les nostres llars han de complir unes propietats per mantenir segures les nostres dades. Parlem de la confidencialitat, integritat i disponibilitat.

- Confidencialitat: és la propietat que impedeix que es divulgue informació a persones o sistemes no autoritzats. Per aconseguir confidencialitat,

s'aconsella l'ús de criptografia, renovar les claus d'accés periodísticament i implementar mecanismes d'autentificació dels dispositius més segurs.

- Integritat: és la propietat que les dades no han estat canviades, eliminades o perdudes de manera intencionada o no desitjada. Els mecanismes de comprovació d'integritat (CRC), signatures digitals i les claus ens prevenen de les dades modificades.
- Autenticació: és el mecanisme per identificar el emissor de la informació. Per a la autentificació s'han d'usar codis o signatures digital. Un mètode molt emprat són les claus simètriques, però, es recomana que aquesta clau siga única per cada dispositiu per evitar utilitzar les mateixes credencials per a diferents dispositius.
- Frescor: del anglès *freshness*, és el mecanisme per verificar que el paquet tramés es nou i no ha estat reutilitzat de anteriors sessions. Per aconseguir aquesta propietat s'utilitzen marques de temps, comptadors o els denominats *nonce* – nombres arbitrari que sols poden ser utilitzats una volta.

3.5 Elements d'un sistema domòtic

En els darrers anys, la fragmentació de la tecnologia i dispositius ha augmentat per falta de estàndards consolidats. La tendència de les empreses és utilitzar solucions pròpies. Tot i això, organitzacions a nivell mundial, sabent els inconvenients d'aquesta situació, tracten de millorar i proposar estàndards de manera col·laborativa.

Perquè la informació viatge entre els dispositius, necessitem d'uns protocols de comunicació estandarditzats. Dispositius, que com ja hem apuntat, poden exercir diferents funcions al nostre sistema. Per entendre els diferents estàndards de comunicació existents i el maquinari i programari necessari per poder implementar un sistema domòtic amb les característiques requerides per aquest TFG, en els següents apartats abordarem en més detall aquests elements.

4 Protocols

Les xarxes domòtiques estan construïdes sobre uns estàndards i protocols per la comunicació entre els dispositius. Aquests estàndards són molt variats i extensos. A continuació es detallen una serie de protocols i d'estàndards molt emprats avui dia i que són oberts, és a dir, qualsevol empresa o públic en general pot fer ús d'aquests i implementant-los.

Cal afegir que, tot i que Z-Wave no és un estàndard del tot obert, fa poc ha comunicat un procés de obertura de part de la implementació que n'era oculta. Junt amb aquesta decisió, grans companyies com Apple, Google, Amazon, Ikea, Samsung o Philips, han anunciat una unió per desenvolupar un nou estàndard per als habitatges intel·ligents, Connected Home over IP, que serà llançat al 2021, i que segur, canviarà totalment l'ecosistema de la domòtica actual.

4.1 KNX

L'estàndard obert KNX és un protocol de comunicacions de xarxa molt utilitzat en domòtica. La seva creació té origen en 1999, quan tres de les solucions que existien a Europa per al control d'habitatges i edificis, decideixen unir-se amb l'objectiu de desenvolupar conjuntament una norma industrial internacional. Aquestes tres associacions -EIBA (*European Installation Bus Association*), EHSA (*European Home Systems Association*) i BCI (*BatiBUS Club International*)- fundaren la *KNX Association cvba* amb sede en Brussel·les, Bèlgica.

KNX fa servir un sistema de bus, d'aquesta manera, tots els dispositius es troben connectats al mateix medi físic i poden intercanviar informació a través d'ell. Aquest fet fa que KNX empri una topologia descentralitzada, i a conseqüència d'això, no és necessària una unitat central de control. No obstant això, també és possible instal·lar-lo per a aplicacions més especifiques. Els

sistemes KNX són estables, segurs, eficients i molt escalables perquè poden arribar a gestionar més de 50.000 dispositius diferents.

Els medis de transmissió definits al protocol són:

- Cable parell trenat (KNX TP). És el medi més utilitzat pel seu baix cost i facilitat d'instal·lació. Sobretot utilitzat per a construccions noves.
- Xarxa elèctrica (KNX PL). Utilitza la xarxa de força existent a un edifici.
- Ràdio (KNX RF).Transmissió per radiofreqüència. Molt útil quan l'ampliació de la xarxa KNX és molt complicada.
- Ethernet (KNX IP): A través de la xarxa *ethernet*, els telegrames KNX es transmeten codificats entre els dispositius participants. KNX IP Secure permet l'autenticació i codificació dels telegrames enviats, d'aquesta manera es pot garantir que els telegrames, amb tunelització o encaminament, no poden ser interceptats o manipulats per aliens.

A nivell de seguretat, aquest protocol presenta vulnerabilitats. El fet de compartir el mitja de comunicació sense xifratge ni autentificació implica que amb accés al bus KNX, podem accedir i gestionar qualsevol dispositiu. Per esmenar aquesta feblesa, KNX IP Secure tracta de afrontar els reptes de seguretat d'aquest sistema. La millora protegeix la comunicació IP entre els dispositius, ampliant el protocol IP per xifrar amb AES-128-CCM tots els missatges i les dades transferides, i CBC-MAC per protegir la integritat dels missatges. La identitat del telegrama es verifica amb l'adreça física. Cada destinatari posseeix una taula (ETS) amb les adreces físiques dels dispositius permesos. I encara més, es dispositius KNX Secure realitzen un seguiment d'atac si els dispositius o la xarxa ha estat sotmesa a atacs de seguretat.

Taula 2. Resumen en matèria de seguretat de KNX-IP secure

Autentificació	**Integritat**	**Confidencialitat**	**Frescor**
Taula ETS amb MAC	CBC-MAC	AES-128	Numero de seqüencia de 6 bytes

4.2 LonWorks

A la dècada del noranta, als Estats Units es va desenvolupar un estàndard per a l'automatització de tot tipus d'infraestructures. LonWorks permet dissenyar sistemes de control distribuïts, escalables i compatibles amb altres sistemes. Admet qualsevol tipologia de xarxa.

LonWorks utilitza per a l'intercanvi d'informació el protocol LonTalk. Aquest protocol és obert i, per tant, està disponible per a qualsevol fabricant. És independent del medi, permetent als dispositius comunicar-se sobre qualsevol mitjà de transport. Els equips es comuniquen de forma directa, sense necessitat de controladors centralitzats. A diferencia de KNX, el protocol accepta topologia de cablejat en anell, a més d'oferir major velocitat de transmissió.

Pel que fa la seguretat, al igual que KNX, a es tracta d'un protocol sense xifratge i que usa mitjans físics. La diferència és, que LonWorks sí usa autentificació del remitent per verificar els missatges. És tracta d'una clau única de 48 bits per cada dispositiu, que s'ha d'assignar al inici de la instal·lació de la xarxa i que tots els dispositius coneixen. Per verificar que els missatges són legítims utilitza un protocol de desafiament-resposta de 64 bits. Tanmateix, la informació es segueix enviant amb text clar, que a pesar de no poder modificar-lo, continua sent llegible.

LonMark International (LMI), organització mundial sense ànim de lucre, ha creat guies per al desenvolupament de productes amb aquest protocol, i proporciona ferramentes, recursos i recolzament per als membres.

Taula 3. Resumen en matèria de seguretat de LonWorks

Autentificació	**Integritat**	**Confidencialitat**	**Frescor**
Clau 48 bits	protocol de desafiament-resposta de 64 bit	-	Numero de seqüencia de 64 bits

4.3 X10

X10 és un protocol de comunicacions per al control remot de dispositius electrònics que utilitza el cablejat elèctric per a enviar la informació entre els dispositius. Tot i que el gran avantatge és el baix cost d'instal·lació, actualment aquest sistemes estan caient en desús a causa de la poca fiabilitat i estabilitat.

Té un número d'identificació de 4 bits únic, el que limita a 16 números d'identificació únics. També té un limitat nombre d'ordres codificat amb 4 bits. El fet de compartir el cablejat elèctric fa que apareguen fallades en la transmissió de la informació i la absència de xifratge obri un porta d'accés a que algú es connecte a la xarxa i controle els dispositius.

Taula 4. Resumen en matèria de seguretat de X10

Autentificació	Integritat	Confidencialitat	Frescor
4 bits	-	-	-

4.4 ZigBee

ZigBee és un estàndard de comunicacions sense fils dissenyat per ZigBee Alliance. Es tracta d'un protocol obert que està basat en l'estàndard IEEE 802.15.4 de xarxes sense fils d'àrea personal (WPAN) i té com a objectiu les aplicacions que requereixen comunicacions segures amb baixa taxa d'enviament de dades i maximització de la vida útil de les bateries.

Funciona a traves d'ones de ràdio de manera bidireccional, i té un abast fins a 75 metres. Aquestes ones de ràdio de baixa freqüència permeten propagar-se més lluny, perquè són menys absorbides pels materials de construcció. Operen a velocitats de 250 Kbps, i es tracta d'un protocol de comunicació multisalt, és a dir, que es pot establir connexió entre dos nodes fora del radi de transmissió, sempre que existeixin nodes intermedis. Aquest fet també incorpora un grau de estabilitat, ja que és independent de la fallada de un node. Les xarxes ZigBee admeten topologies d'estrella, d'arbres i de malla.

A una xarxa ZigBee poden existir tres tipus de dispositius diferents:

- El coordinador s'encarrega de controlar la xarxa i les rutes de comunicació.
- Els encaminadors, que és el únic tipus no obligatori, interconnecta els nodes amb el coordinador, i ens ajuda a crear la topologia de xarxa desitjada.
- Els dispositius finals sols reben i envien informació amb el node pare. Aquests aparells poden ser tant actuadors com sensors.

Cal remarcar que els dispositius ZigBee no són adreçables per IP, els usuaris han de connectar-los amb una passarel·la per poder connectar-s'hi. Més que un protocol, és una pila de protocols que està format per diferents capes independents l'una de l'altra, cadascuna amb una funció especifica. Entre les empreses que utilitzen aquest sistema trobem Amazon Echo, Philips, OSRAM, Samsung Electronics, IKEA o Toshiba.

Quant a seguretat, ZigBee proporciona mecanismes de control d'accés (autentificació), xifratge (criptografia de claus simètriques AES-128) i integritat de missatges(CCM). Actualment, la seguretat està centralitzada en el coordinador de la xarxa. Utilitza tres tipus diferents de claus en diferents parts de la pila per associar-se a una xarxa: clau mestra, la qual és la clau inicial i s'assigna durant la instal·lació; clau d'enllaç, que xifra la comunicació punt a punt a nivell d'aplicació i és diferent per a cada parella de dispositius; clau de xarxa, la qual s'utilitza a nivell de xarxa i és comuna a tota la xarxa.

Si hem de nombrar alguna debilitat, donada la poca potència d'alguns dispositius i si aquests funcionen a bateria, poden esgotar la carrega amb menys temps que altres sistemes. A més a més, les claus es guarden en la memòria del dispositiu, i tenint accés físic i amb programari específic podrien accedir a les dades. En qualsevol cas, com hem pogut veure, la seguretat és una de les fortaleses de ZigBee.

Taula 5. Resumen en matèria de seguretat de ZigBee

Autentificació	Integritat	Confidencialitat	Frescor
AES-CCM	AES-CCM	AES-128 CTR	Contador+nonce

4.5 Z-Wave

Protocol tecnològic sense fils per a la comunicació dels diferents aparells electrònics d'un habitatge. Atès al baix cost, fiabilitat i facilitat de instal·lació i configuració, aquest sistema és un dels més utilitzats a nivell mundial

Empra ones de radiofreqüència de baixa potencia que suporta xarxes en malla sense la necessitat d'un node coordinador. Com ZigBee, opera en un espectre radioelèctric diferent a WI-FI o Bluetooth, i per tant transparent a les interferències d'aquests.

La seva topologia és una xarxa de malla encaminada que permet la comunicació de dos dispositius sense que es trobin dintre de l'abast de cobertura. Aquest fet facilita la implementació del sistema per a usuaris novells.

Z-Wave Plus és la nova generació d'aquest sistema, què incorpora una nova capa d'autenticació que pretén evitar que els pirates informàtics s'apoderin del control de dispositius no segurs o mal protegits. A més, hi augmenta la velocitat, l'abast i redueix el consum.

Cada dispositiu té un identificador de node i cada xarxa Z-Wave està identificada per un identificador de 32 bits que és assignat a cada dispositiu pel controlador principal quan el dispositiu s'afegeix a la xarxa. El xifratge que utilitza és AES-128, però a la versió S2 del protocol de seguretat, es va afegir un nou sistema per autentificar els dispositius. Anomenat DSK, es tracta d'un parell de claus pública-privada, on la clau pública de 32 bytes no s'envia completa per la xarxa, sinó que els dos primers bytes s'han de assignar de manera manual bé per mitja d'un pin o amb un codi QR.

Quant a empreses que implementen aquest sistema tenim a Fibaro, que en els darrers anys ha obtingut gran rellevància a Europa per la gran varietat de productes, qualitat i preu.

Taula 6. Resumen en matèria de seguretat de Z-Wave

Autentificació	Integritat	Confidencialitat	Frescor
CBC-MAC	CBC-MAC	AES-128 OFB	Nº de seqüencia de 64 bits

4.6 MQTT

MQTT, Message Queue Server Telemetry Transport, és un protocol de xarxa de publicació-subscripció lleuger per a la comunicació entre dispositius. Usualment s'executa sobre TPC/IP, tot i que es pot executar sobre altres protocols de xarxa que proporcionen connexions bidireccionals ordenades i sense pèrdues. Està orientat per al Internet de les coses (IoT). Administrada per OASIS, actualment es troba en la versió 5a.

Els actors principals són els agents MQTT, que són els servidors que reben els missatges dels clients, per a després encaminar-los al clients destinataris. Un client pot ser qualsevol dispositiu que implemente MQTT i es connecte a un agent a través de la xarxa.

MQTT ofereix diverses formes d'autentificar-se, bé siga amb usuari-contrasenya o certificat X.509. Es recomana el ús de X.509, que és un estàndard de clau pública més segura que el identificador del client. Per utilitzar aquest sistema és necessari una autoritat certificadora. A més de la autentificació, MQTT autoritza als clients, ja sigui per subscriure's o publicar en temes, d'un sistema basats en rols -més abstracte- i llista de control d'accés-més específic.

Taula 7. Resumen en matèria de seguretat de MQTT

Autentificació	Integritat	Confidencialitat	Frescor
Signatura digital	Signatura digital MAC suma de verificació	TLS	Marca de temps

5 Maquinari

Podem definir tres gran grups de maquinari necessari:

- Controladors: equip central del sistema que controla tots els dispositius, sensors i actuadors. Fan ús de protocols de comunicació per a transmetre la informació.
- Sensors: S'empren per mesurar i enviar informació al controlador domòtic. Els sensor més usuals són els de temperatura, de moviment o de portes i finestres, però hi ha de molts tipus.
- Actuadors: dispositiu que realitza una acció emesa pel controlador arran de les dades enviades pels sensors.

Pel que fa a maquinari ens centrarem amb el maquinari necessari per implementar els controladors, i que ens permeta configurar un sistema a la nostra elecció, sense dependències d'empreses alienes.

5.1 Arduino

Arduino és una placa de desenvolupament que incorpora un microcontrolador junt amb determinats ports GPIO. És un maquinari lliure i posa a disposició un IDE també de codi obert. Els microcontroladors es programen amb C i C++. Actualment hi ha una variada oferta de plaques Arduino, segons la necessitat de pins, mida o memòria.

Arduino s'ha popularitzat molt per a projectes casolans o investigació, degut al seu preu reduït i la seva versatilitat. Gràcies a la seva llicencia, han sorgit unes plaques derivades d'Arduino, com les fabricades per SainSmart, Smart Citizen o Makey Makey.Quant a IDE's, té un programari propi disponible per a Windows, MacOS i Linux. Com no podia ser d'altra forma, el entorn de desenvolupament és de codi obert.

Avui dia, a la tenda oficial, a banda de plaques de desenvolupament o d'expansió, disposen de diversos kits per a la Internet de les coses. Com el *IoT*

Oplà que permet afegir connectivitat a dispositius de la llar com llums, estacions meteorològiques, termòstats o control de plantes d'una manera senzilla i poder controlar-ho per mitja del mòbil.

5.2 Raspberry Pi

Raspberry Pi és un computador de mida reduïda que disposa pins d'entrada de propòsit general. No és un maquinari lliure, ja que la fundació amb el mateix nom té la propietat de disseny enregistrada. Tot i això, es permet el seu ús lliure per a tothom. Posseïx un sistema operatiu propi basat en Debian, Raspbian, que sí és de codi obert. Encara que, existeixen gran quantitat de sistemes operatius complets que funcionen en aquest computador.

Per norma general, Raspberry són plaques més potents que Arduino i existeixen diferents versions. Les més utilitzades són Raspberry Pi 3 Model B i la versió B+ amb més potencia. Al juny de 2019 es va llençar la ultima versió Pi 4 Model B. També tenen disponibles diferents accessoris per ampliar les funcionalitats del computador com ara càmeres targetes d'expansió per a televisió. Es distribueixen sense carcassa ni font d'alimentació i utilitzen com a disc dur targetes SD que tampoc són incloses.

La comunitat al voltat de Raspberry en immensa amb fòrums oficials en diferents idiomes. Fins i tot publica mensualment una revista per divulgar informació diversa i exemples de projectes. I al gual que Arduino, han nascut altres plaques basades en Raspberry Pi com Orange Pi o Banana Pi.

5.3 Asus Tinker Board

La multinacional Asus també fabrica plaques *open-hardware* de desenvolupament molt similars a Raspberry Pi. Aquestes plaques ofereixen més potencia i qualitat a totes les parts del sistema, però, amb un preu més elevat. Disposa d'un sistema operatiu basat en Debian, anomenat TinkerOS, extremadament lleuger i adaptat a aquesta placa.

5.4 Odroid

Família d'ordinadors mono-processador i tauletes del fabricant Harkernel, companyia especialitzada en maquinari lliure. Malgrat tot, Odroid conté parts del disseny amb copyright. Com a sistemes operatius disponibles, a més d'Android, té una gran varietat de sistemes Linux.

6 Programari: Controladors Domèstics

Les unitats encarregades de rebre la informació del sensor, emmagatzemar-la, gestionar-la i finalment emetre les ordes als actuadors, són els controladors domòtics.

La implantació de sistemes domòtics en les nostres llars s'ha incrementat en els últims anys. Aquest fet és degut a la popularització de la tecnologia que ens proposen les grans corporacions com Apple, Google, Amazon o Samsung. Actualment, però, no ofereixen una interoperativitat amb totes les diferents tecnologies existents. Sembla que la experiència sempre és més complaent quan es fa ús de dispositius del mateix grup empresarial.

Per evitar aquesta manca de interoperabilitat, i oferir control als usuaris que es preocupen per la seguretat i privacitat, existeixen unes plataformes de codi obert que ens simplifiquen la tasca de configuració de tot l'ecosistema. A continuació, s'exposen les més populars amb una gran comunitat al voltant.

6.1 Calaos

Calaos és una solució completa per a la domòtica de les nostres llars. La empresa francesa va fer fallada a 2013 i va alliberar el codi, i una comunitat de programadors van continuar el desenvolupament del sistema.

Disposa de sistema operatiu propi, basat en Linux, programari amb interfície web, pantalla tàctil o aplicacions natives per a dispositius mòbils, i té suport per al maquinari Raspberry Pi. La documentació, en francès, no és molt exhaustiva, i el desenvolupament i l'ampliació de funcionalitats és molt lent.

6.2 Domoticz

Domoticz és un sistema de codi obert i gratuït, escrit en C++. El maquinari que suporta és variat, des de sistemes operatius com Windows, MacOS o Linux, fins a sistemes encastats com Raspberry Pi.

Disposa de connectors natius per a les interfícies més demandades com X10 o ZWave, i permet la integració de mòduls de tercers. Té suport *multiidioma*, entre els quals, espanyol i català.

Inclou el seu propi servidor web incrustat, per a aconseguir execució eficient i evitar dependències, amb compatibilitat HTTP/SSL. Ofereix una interfície web HTML5 escalable, què facilita la accessibilitat per mitja del qualsevol navegador modern, tant en ordinadors com en telèfons intel·ligents. Amb 'fail2ban' aconseguim un mur de protecció per a connexions que realitzen diversos intents fallits d'autentificació. A més, té diverses aplicacions per a dispositius mòbils, com ara Android i iOS. Les aplicacions oficials també són de codi obert i no és necessària una connexió a Internet per funcionar.

Les automatitzacions s'assoleixen amb Blockly, més orientat a usuaris no avançats, on la codificació es realitza engranant blocs de codi de forma visual. Un altre mode disponible és Lua, un llenguatge de programació d'*scripts* per a usuaris més avançats.

6.3 OpenHAB

Open Home Automation Bus, openHAB, és una plataforma de codi obert desenvolupat amb Java i utilitza OSGi per a la modularitat. Aquest sistema està disponible per a una gran quantitat de sistemes, tant sistemes operatius com sistemes encastats. També disposa d'aplicacions oficials per a dispositius mòbils *Android, iOS i Windows.*

Per facilitar-ne la instal·lació i configuració en dispositius Linux, han implementat una distribució pròpia anomenada OpenHABian, que conté l'últim programari de OpenHAB amb totes les configuracions estàndard i dependències necessàries. També, té una comunitat molt activa que resol els dubtes que puguin sorgir, i una documentació molt extensa.

Amb una arquitectura totalment modular, ofereix una extensa quantitat de mòduls per a interactuar amb els dispositius, abastant la major part dels dispositius disponibles. Respecte a possibilitat de controlar el sistema a través

de la veu que siguin *offline*, els únics sistemes disponibles són PicoTTS i MariTTS.

Quant a les automatitzacions, són escrites amb Xtend, un dialecte de Java, amb una sintaxis més concisa. Cada regla es divideix en dues seccions: la secció *when*, on s'afegeixen els esdeveniments que llançaran la regla; i la secció *then*, on es detallen totes les accions a realitzar quan es llença la regla.

En matèria de seguretat, la comunicació amb el servidor web es realitza amb SSL amb un certificat auto-firmat de 256 bits ECC únic per a cada servidor. No obstant això, no admet autentificació d'usuaris amb SSL, i aquest fet pot ser una porta d'entrada de atacs externs.

6.4 Home Assistant

Plataforma *open-source* per a la automatització total d'un habitatge, escrit en Python3. El maquinari compatible són els sistemes Raspberry Pi 3 o superior, TinkerBoard, i algunes plaques Odroid i Intel NUC. Té disponibles gran quantitat de mòduls per la interacció amb distintes plataformes, serveis o dispositius. Actualment disposa de més de mil sis-cents mòduls per a aconseguir una integració amb la major part dels dispositius més populars que existeixen. També habilita una *Rest API* per a desenvolupadors.

Home Assistant no utilitza cap connexió a Internet, totes les dades es tracten de manera local. A més, compta amb aplicacions per a Android i iOS també de codi obert i una comunitat molt gran, tant de desenvolupadors com de suport. Té accés multiusuari, amb possibilitat de afegir doble anell de seguretat amb Multi-factor Authentucation (MFA). També té suport a SSL i Fail2Ban i recomanen fer ús d'un *proxy* intermediari per aconseguir major seguretat.

Pel que fa als sistemes de veu compatibles, a banda dels més comercials, com Alexa, Google Assistant o HomeKit, hi tenim disponibles Almond, MyCroft o Snips.

6.5 ioBroker

Aquesta plataforma de codi obert està escrita amb Node.js. Aquest fet fa que el sistema pugui ser executat en qualsevol sistema amb JavaScript habilitat. Les automatitzacions es realitzen també amb JavaScript o, per a usuaris novells, amb la variant Blockly. Té una construcció modular i una gran quantitat de mòduls que amplien les funcionalitats del sistema. Pots habilitar el xifrat HTTPS i l'autentificació d'usuaris, encara que no estan actius per defecte.

Actualment hi té disponibles més de tres-cents d'aquests adaptadors, entre els quals trobem els protocols ZigBee i Z-Wave, de dispositius Philips i Samsung, o d'aplicacions com Spotify i Kodi. D'assistent de veu té disponible Sonus, projecte de codi obert, que fa ús de Google Assistant per obtenir les respostes.

6.6 WebThings

WebThings és un projecte que està desenvolupant Mozilla per al Internet de les coses. Surt de la unió de dos projectes anteriors: WebThings Framework i WebThings Gateway. Aquesta opció tracta de simplificar tant la creació de nous dispositius IoT com la configuració i gestió centralitzada del sistema.

Està escrit en JavaScript, i ofereix un interfície d'usuari basada en web, sobre Node.js per a la gestió. Per a la programació de components podem fer ús de diversos llenguatges de programació com Python, Java o JavaScript.

Disposa de complements per ampliar funcionalitats tant pròpies com de tercers. La de control de veu l'ofereix DeepSpeach, un STT sense connexió també de Mozilla. Actualment es pot instal·lar en una RaspBerry Pi i es poden utilitzar els protocols més utilitzats com ZigBee, Zwave, WIFI o BlueTooth. Tot i trobar-se en un estat molt prematur, la comunitat és molt activa.

L'objectiu de Mozilla és promoure els estàndards oberts emfatitzant la seguretat i la protecció de dades. Per tant, les connexions amb el navegador estaran xifrades amb SSL i té disponible l'autentificació de multiusuaris.

6.7 Altres Sistemes

6.7.1 PiHome

Projecte de codi obert basat en el dispositiu Raspberry Pi, per al control de la calefacció de l'habitatge. Escrit amb Python, utilitza un servidor Apache com interfície amb el usuari i una base de dades MySQL per guardar la informació. Es troba en una fase primerenca de desenvolupament, però es manté activa i funcional. La documentació és molt limitada i dispersa.

6.7.2 Jeeddom

Sistema parcialment de codi obert, compatible amb els protocols més utilitzats actualment, disponible sols per a sistemes Linux. Jeedom té una enginyosa *Jeedom Market* per ampliar les funcionalitats del sistema i té apps natives per a Android i iOS. Posa a disposició dels clients diferents lots, gratuïts o de pagament, segons les funcionalitats que incorpora. Un dels serveis que ofereix és el assistent de veu, amb l'ajuda dels assistents de Google o Amazon.

7 Programari: Assistents Personals

Un assistent virtual intel·ligent o assistent personal intel·ligent és un programari que ajuda a les persones a realitzar tasques o serveis per mitja de comandes o preguntes i té com a mètode d'entrada la veu o text.

Aquest sistema s'ha esdevingut una excel·lent forma d'interactuar amb els dispositius de les nostres cases o oferir informació puntual amb una mínima interacció. Per aconseguir proporcionar respostes acurades als usuaris, aquestes plataformes utilitzen una connexió a uns servidors externs on s'emmagatzemen les dades personals. Amb aquestes dades dels usuaris es creen perfils sociodemogràfics, d'interessos i preferències personals per utilitzar-los amb campanyes publicitaries.

Pel que fa als assistents de veu hem de diferenciar diversos components que funcionen com a petites capes independents i donen resposta a funcions específiques necessàries:

- Wake Word Spotter: aquesta capa s'encarrega d'escoltar de manera continuada fins que sent la paraula 'despertador', la qual activarà la gravació de veu.
- Speech To Text (STT): component que converteix els sons de entrada a text. Alguns d'aquests sistemes requereixen d'un període de formació perquè el sistema analitza la veu específica de la persona per afinar el reconeixement i reduir la taxa de fallades.
- Intent Parser: capa que s'encarrega de extraure el text del STT i determinar l'acció que hi ha que realitzar. Aquestes accions són denominades intencions.
- Text To Speech (TTS): component invers de STT. Una volta acabada la tasca a realitzar determinada per la intenció, l'assistent pot oferir una resposta o sortida i TTS és el component que la convertirà en veu.

Com en aquest TFG estem tractant de oferir un sistema sense fer ús de servidors externs, estudiarem assistents de veu on tots els components que són necessaris compleixen aquest requeriment. Som conscients que al prescindir de la connexió a servidors externs, ens trobarem amb sistemes més simples i menys eficients, ja que aquests servidors realitzen una tasca computacional elevada i contenen gran quantitat d'informació que no es pot emmagatzemar en un dispositiu amb les característiques que estem estudiant..

7.1 Almond/Ada

Almond és un assistent de codi obert de la plataforma Open Virtual Assistant Lab Program (OVAL) de la universitat de Stanford. Aquest programari tracta de democratitzar la tecnologia d'assistents virtuals, protegir la privadesa dels usuaris i mantenir un accés obert al coneixement. Admet autentificació amb OAuth 2.0 o amb usuari-contrasenya.

Es necessita el component LUInet, un model neuronal també de codi obert que pot comprendre les comandes del assistent virtual. Els models neuronals apareixen com a resposta a problemes complexos i imiten el funcionament de les xarxes neuronals del humans. A la pràctica, el model rep uns paràmetres d'entrada, i el model ha de combinar-los per predir un resultat. LUInet està disponible a través d'un servici web o com a un servidor privat local.

Però Almond sols funciona amb entrades de text, i genera també text com sortida. Per tal d'oferir funcionalitats amb la veu és necessari agregar-li un altre component i Ada és la part del sistema que realitzarà les tasques amb la veu. Aquest duet es troba a una fase inicial de desenvolupament, però amb el recolzament de Standford i de la plataforma Home Assistant pot aconseguir bons resultats en un futur proper.

7.2 Mycroft

Assistent de veu format per un conjunt de tecnologies de programari lliure. Tot i que als inicis tenia una llicencia *copyleft*, actualment es distribueix sota una llicència permissiva Apache.

Entre les tecnologies existents a Mycroft destaquem Mimic com a TTS, Common Voice de Mozilla com STT i Precise com a oient de paraules de vigília. Mycroft necessita d'un servidor extern descentralitzat per al processament i per donar una resposta més acurada a les comandes. Actualment, els desenvolupadors treballen per habilitar un servidor personal amb menys funcionalitat per donar resposta a tot tipus de situacions. Altrament, donada la construcció modular de Mycroft, és possible utilitzar mòduls tant de STT com de TTS que no requereixen connexió a Internet.

Mycroft està disponible per a sistemes Linux, i té versions especifiques per a Raspbery Pi i Docker. L'aplicació d'Android, encara no oficial, emprà el TTS de Google. A la tenda de Mycroft es poden trobar altaveus intel·ligents que incorporen el sistema ja configurat. Actualment treballen en la versió 3 del sistema en fase de prototips de maquinari.

7.3 Rhasspy

Rhasspy és un conjunt de serveis d'assistent de veu de codi obert els quals es troben desconnectats d'Internet. Amb integracions amb NodeRED i Home Assistant, té suport multi-idioma, entre els quals es troba el espanyol i català.

Una volta rep la comanda de veu la transforma en un esdeveniment JSON, que el compararà amb les comandes prèviament incloses i executarà les comandes configurades. Tant els *websockets* com l'API HTTP són utilitzats per a interactuar amb serveis externs i ambdós admeten la capa de seguretat SSL . La instal·lació es pot realitzar per mitja de Docker i sobre sistemes Linux. Per a distribucions basades en Debian, ofereix un paquet *precompilat*.

Actualment no es recomana Rhasppy per a usuaris sense coneixements, ja que es troba en fase de desenvolupament. Malgrat això, el codi rep actualitzacions constantment amb una extensa documentació i un panell de millores futures i problemes detectats per usuaris.

8 Sistema seleccionat

A continuació es detallaran els elements del sistema i els motius pels quals han estat seleccionats. A més del maquinari i programari que hem descrit anteriorment, es consideraran també, tot els elements que seran necessaris per implantar el sistema. Continuant amb els propòsits inicial d'implementació, es seleccionaran sistemes amb codi obert i que ofereixen mecanisme de seguretat i protecció de les dades personals.

Per a la implementació es descartaran dispositius que necessiten de manipulació elèctrica i s'utilitzaran xarxes sense fils donada la facilitat d'instal·lació a un habitatge ja construït. Tots els dispositius es comunicaran amb un controlador domèstic, que serà qui emmagatzemi les dades, prenga les decisions i envie les instruccions als dispositius. Aquest sistema central haurà d'utilitzar mecanismes de seguretat bàsic.

8.1 Protocol

Avui en dia, la lluita dels protocols segueix sent força complicada. Els protocols més habituals en domòtica són ZigBee i Z-Wave. Ambdós fan servir, com hem comentat anteriorment, ones de radio de baixa freqüència. Aquests s'han estès considerablement front a sistemes WIFI, perquè eviten la saturació de la xarxa quan es connecten un nombre considerable de dispositius. A més, cap dels dos utilitzen protocols IP, el que fa que no es connecten directament a Internet i ens ofereixen millor privacitat.

Com analitzàrem, el protocol Z-Wave no és completament de codi obert, i a més, és necessària una llicencia d'us per desenvolupar dispositius amb aquest protocol. Per una altra banda, el projecte Connected Home over IP que tracta d'augmentar la compatibilitat del productes per a la llar, es troba encara en una fase inicial de desenvolupament.

Per aquesta raó, creiem convenient seleccionar el protocol ZigBee per a la nostra implementació, a més, té mecanismes de seguretat com xifratge de les

comunicacions i autentificació segura. Un altre aspecte interessant és que els dispositius es converteixen en encaminadors i augmenten la cobertura de forma transparent. I al ser gratuït, s'aconsegueix una reducció en els dispositius que l'implementen i cada volta més, està recolzat per les grans companyies.

Per a la creació d'una xarxa ZigBee necessitem una passarel·la que es comunique amb la plataforma domòtica. Les passarel·les són oferides per empreses que implementen dispositius amb aquest protocol. Ara bé, la interoperabilitat entre dispositius de diferents empreses no està sempre assegurada, ja que aquestes prioritzen les connexions amb dispositius propis. Sols algunes amplien el ventall de dispositius compatibles amb acords privats amb altres empreses del sector. També existeixen al mercat opcions per fer front a la incompatibilitat, com el USB ConBee amb una gran diversitat d'aparells compatibles, però tenen un cost econòmic elevat i no són de codi obert.

Nosaltres ho implementarem amb un xip CC2531 de Texas Instruments, el qual amb *flashing* li instal·larem un microprogramari perquè funcione com a coordinador ZigBee. CC2531 és un sistema en xip (SoC, del anglès *system on a chip*) amb port USB 2.0. El xip integra, entre altres coses, un transceptor, una memòria programable de 256 Kb i una memòria RAM de 8 Kb. Per al *flashing* necessitarem també un depurador i programador de sistemes RF sobre xips, CC-Debugger. Utilitzarem Z-Stack versió 1.2 de coordinador (el procés d'instal·lació està detallat a l'annex 'Coordinador ZigBee'). D'aquesta manera ens assegurarem la compatibilitat en diferents dispositius de diferents marques amb un baix cost. Cal afegir, que el microprogramari és de codi obert, i per tant compleix els requisits d'aquest treball.

La tipologia de xarxa seleccionada per a ZigBee és centralitzada, atès que actualment no tenim molts dispositius, i donades les dimensions del habitatge, no tindrem problemes de cobertura. Malgrat tot, si en un futur la xarxa augmentara tampoc veuríem minvada la qualitat i velocitat.

No es considera que el sistema sigui accessible externament, sols internament. Malgrat tot, la comunicació es realitzarà via dos protocols: per terminal amb SSH, i amb HTTPS per a la connexió amb les interfícies web.

8.2 Maquinari

Tant Arduino com Raspberry Pi són sistemes molt interessants que compleixen en els requisits per aquest projecte. En cas de Raspberry requereix d'un sistema operatiu per poder funcionar, el qual ha de ser un Linux que podrem adaptar a les nostres necessitats i afegir-li altres funcionalitats si ho desitgem. La capacitat de càlcul és superior a Arduino i té integrada tant la connexió per WIFI, Bluetooth o *ethernet* a la placa.

Malgrat tot, considerem prioritari l'ús de Raspberry, perquè la gran majoria de assistents personals s'implementen en aquest maquinari i pot oferir millor rendiment amb tot el conjunt de programari que volem implementar. En aquest cas, farem ús de una Raspberry Pi 3B+ amb un targeta microSD de 16GB i font d'alimentació. A més, per demostrar interoperabilitat del sistema amb distintes tecnologies i empreses, seleccionarem un pocs dispositius diversos:

- Xip CC2531 i un CC-Debugger
- Micròfon + Altaveu USB
- SONOFF SNZB-02 ZigBee. Sensor de temperatura i humitat
- Bombeta Ikea Tradfi E27
- Endoll Ikea Tradfi

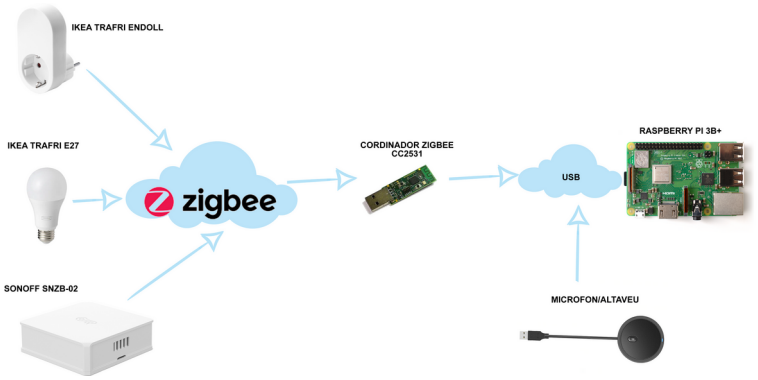

Figura 2: Diagrama del maquinari i protocols del sistema

8.3 Controlador domèstic

Tant Domoticz, OpenHAB i Home Assistant ofereixen sistemes estables que ens faciliten la connexió amb tots els dispositius. Però creiem convenient descartar Domoticz perquè la integració de tots els sistemes seleccionats pot presentar problemes de comunicació i configuració. I en aquests entorns on els sistemes es troben en fase de desenvolupament, la comunitat que hi té al voltant adquireix molta importància per poder resoldre els problemes i per tant hem d'optar per una quant més gran i activa millor.

Per una altra part, el ritme de desenvolupament de OpenHab és més lent. Aquest sistema té un procés d'incorporació de millores i de dispositius suportats més exhaustiu que el converteix en un sistema més estable però amb menys compatibilitats amb dispositius més moderns. Al seu torn, Home Assistant sembla tenir millor integració amb sistemes de reconeixement de veu i parlem d'un sistema complet que inclou una distribució Linux personalitzada, construït amb Buildroot optimitzada per a l'execució de la plataforma en sistemes encastats.

Hem d'afegir que la plataforma de Mozilla WebThings és molt prometedora, però, a ara per ara no ofereix compatibilitat amb molt dispositius. I a més a més, els coordinadors ZigBee que són compatibles avui en dia són tots de codi privatiu i no dona suport al xip que hem seleccionat, el CC2531.

Llavors, la plataforma seleccionada ha estat Home Assistant, que a més de complir amb aspectes d'accés al codi, no empra cap element fora de la nostra xarxa, ofereix sistemes de protecció d'atacs i té un bon rendiment amb el maquinari. Sense oblidar-nos de la gran comunitat al voltant de la plataforma i que ofereix més opcions pel que fa a assistents personals.

Per a la instal·lació del assistent (Annex: Instal·lació de Home Assistant) ens valdrem de la versió Home Assistant Supervisor, anteriorment anomenada Hass.io. Aquesta versió proporciona el Home Assistant complet, el component Supervisor que ajuda a gestionar les actualitzacions i implementa la botiga que ofereix una gran quantitat de complements per ampliar les funcionalitats, com

ZHA (ZigBee Home Automation), el qual afegirem per a la gestió dels nostres dispositius ZigBee. Caldrà de Docker, un projecte de codi obert que virtualitza a nivell de sistema operatiu per lliurar el programari en els anomenats contenidors.

8.4 Assistent Personal

Tot i que el duet Almond/Ada és el sistema seleccionat per Home Assistant per oferir un assistent virtual als seus usuaris. Tanmateix, a l'actualitat està en una fase prematura de desenvolupament i, conseqüentment, encara hi ha components que s'executen al núvol. Al encara més, sols suporta l'anglès.

L'alternativa sembla ser Rhasspy, que també es troba amb un desenvolupament poc madurat, però es capaç de treballar completament sense utilitzar Internet. I encara més, presenta aspectes molt interessants, com diferents eines en cada secció del programari perquè l'usuari pugui seleccionar el més apropiat segons les seves necessitats. Ens referim sobretot als components TTS i STT. També té un component anomenat Wake Word, que evita la gravació constant de l'àudio i evita un ús excessiu d'energia. Altrament, Rhasspy ofereix diversos idiomes a part de l'anglès i té una documentació prou extensa.

La instal·lació,com Home Assistant, es realitzarà dintre d'un Docker. Amb un navegador podrem accedir a la interfície web, la qual permet configurar tots els components del sistema i entrenar el motor de una forma senzilla i intuïtiva.

8.5 Altre programari:

Com a sistema operatiu utilitzarem Raspberry Pi OS (Annex: instal·lació Raspberry Pi OS), anteriorment Raspbian, sistema oficial per a Raspberry. És una distribució GNU/Linux basat en Debian que ofereix tres opcions distintes d'instal·lació. Donada la carrega que suportarà la Raspberry Pi amb tot el programari necessari i atès que no necessitem entorn gràfic en cap cas, navegadors o reproductors de vídeo, hem decidit utilitzar la versió Raspberry Pi OS Lite.

Instal·larem Fail2ban com a escàner per a la prevenció d'intrusos donada la seva lleugeresa i Iptables com a tallafocs per millorar la seguretat. Com hem comentat anteriorment, necessitarem de SSH per a la connexió remota i xifrada amb el servidor i de Docker com a contenidor del programari.

Un contenidor com Docker automatitza el desplegament d'aplicacions proporcionant una capa addicional d'abstracció. Els contenidors empaqueten totes les dependències necessàries per a executar-se amb l'avantatge de poder ser replicat, modificat o reutilitzat fàcilment. Els contenidors no necessiten virtualitzar tot un sistema operatiu per executar-se i per aquest motiu, redueixen la mida de les aplicacions i ofereixen millor rendiment. Altres beneficis de Docker són l'aïllament de la comunicació entre els contenidors i la separació dels processos no dependents per poder executar-los en paral·lel i així augmentar la compatibilitat i manteniment dels diversos sistemes.

Per facilitar l'administració de Docker farem ús de Portainer, també de codi obert. És una aplicació web creada al 2013 per gestionar tots els aspectes referents tant dels contenidors Docker com de Kubernetes. La mateixa aplicació és un contenidor Docker (Annex: instal·lació Docker i Portainer).

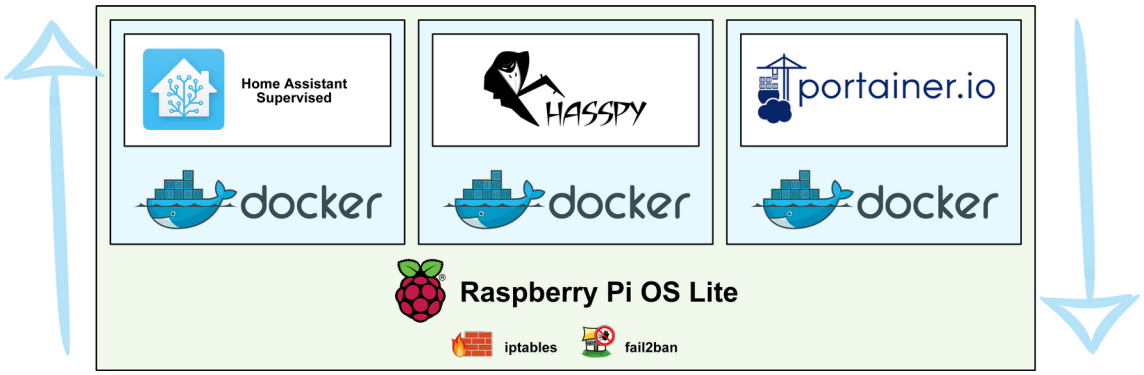

Figura 3: Diagrama de la pila de programari

9 Home Assistant

Una volta instal·lada la plataforma domòtica podem accedir a la interfície web per mitja d'un navegador amb la direcció http://direcció_ip:8123, on cal substituir *direcció_ip* per la IP assignada a la nostra Raspberry. El camp 8123 es refereix al port d'escolta que fa servir Home Assistant. Per mitjà de la interfície web poden configurar gran quantitat d'integracions i de dispositius. Irremeiablement, també haurem de accedir als arxius de configuració al directori especificat a la instal·lació per obtenir configuracions més avançades.

9.1 Arxius de configuració

L'arxiu principal és *configuration.yaml* i conté les integracions que es carregaran al sistema. Com podem apreciar, l'extensió del arxiu indica que utilitza el format YAML, YAML Ain't Markup Language, un llenguatge per a la serialització de dades llegible per humans. És molt utilitzat per arxius de configuració amb format de llistes de parells clau/valor.

A l'arxiu de configuració de Home Assistant es configuren els serveis i dispositius, les automatitzacions i els guions. Existeix l'opció de dividir la configuració entre els diferents arxius com *groups.yaml, automations.yaml* i *scripts.yaml,* molt útils quan el arxiu incrementa en mida considerablement.

Cada dispositiu domòtic necessita d'una entrada en el arxiu per poder llegir els estats i/o poder modificar-los. Per facilitar aquesta tasca, l'assistent té un component de descobriment automàtic que permet descobrir molts dispositius i serveis disponibles a la xarxa.

Donat que l'arxiu principal de configuració és un arxiu de text pla, i per tant llegible per a tot aquell que té accés, les dades sensibles es troben en perill. a l'arxiu *secrets.yaml* és on declarem dades privades que necessitem usar en altres arxius com contrasenyes, claus API i dades de localització. Les declarem aquí de manera centralitzada perquè totes les dades sensibles estiguin en un sol lloc i no repartits per múltiples arxius.

Taula 8. Exemple de configuració de l'arxiu secrets.yaml

secrets.yaml	configuration.yaml
http_password: pass1234 *url_web: https://tfg:8123*	*base_url: !secret http_password* *api_password: !secret url_web*

Aquesta funció és molt útil quan es vol compartir l'arxiu de configuració i evitar que les nostres dades privades siguin accessibles, però les dades segueixen estan guardades amb text pla, només que a un altre arxiu. La forma alternativa és utilitzar un anell de claus.

Un altre element important és el tauler de control anomenat Lovelace. Aquest tauler ens facilita la personalització del elements visuals que trobarem a l'entorn visual. Per defecte és gestiona per la interfície web, si es desitja agafar el control utilitzant YAML, és necessari afegir la entrada 'lovelace' a l'arxiu de configuració general i crear un arxiu nou anomenat *ui-lovelace.yaml* que contindrà tots els elements que configurem.

Taula 9. Exemple de configuració de l'arxiu ui-lovelace.yaml

ui-lovelace.yaml	configuration.yaml
title: Casa *views:* *-title: Home* *cards:* *- type: markdown* *content: Aço és custom* *tittle: Títol*	*Lovelace:* *mode: yaml*

A *customize.yaml* es pot personalitzar l'aspecte per exemple el nom o la unitat de mesura dels dispositius prèviament configurats. Per poder utilitzar aquesta configuració, igual que Lovelace, cal afegir l'entrada a l'arxiu de configuració general.

Taula 10. Exemple de configuració del arxiu customize.yaml

customize.yaml	configuration.yaml
Light.12341234: *friendly_name: Llum entrada*	*Homeassistant:* *customize: !include customize.yaml*

9.2 Automatitzacions

Les automatitzacions realitzen tasques de forma automàtica, sense participació ni interacció amb el usuari. Es divideixen es tres parts: disparadors (*triggers*), condicions (*condition*) i accions (*action*). Els disparadors són els esdeveniments que desencadenen les automatitzacions. Les condicions són opcionals i limiten que una regla funcione només en uns casos específics, per exemple un tram horari concret. Per últim, les accions són les tasques que es realitzaran en un o diversos dispositius quan es dispara la automatització i es compleixen totes les condicions.

Taula 11. Exemple de configuració de l'arxiu automations.yaml

automations.yaml	configuration.yaml
# Exemple automatització *- alias: 'Bon dia'* *trigger:* *- platform: sun* *event: sunrise* *condition: state* *entity_id: light.cuina* *state: on* *action:* *- type: turn_on* *device_id: mediaplayer.radio*	*automation: !include automations.yaml*

9.3 Guions

Els *scripts*, o guions, són una seqüència d'accions que el controlador executarà, i a diferencia de les automatitzacions, no tenen desencadenants ni condicions. L'estructura bàsica de sintaxi és una llista de parells clau/valor que contenen les accions a realitzar. Els guions són molt útils quan una automatització conté moltes accions a realitzar. En aquest cas, és recomana crear un guió i cridar-lo des de l'automatització. Un altre cas d'ús és quan una seqüencia de comandes es repeteix en diversos automatitzacions, i per tant seria interessant fer la crida en tots els llocs al guió. Aquests casos d'us ens permeten tenir un sistema més clar i més fàcil de mantenir. Els guions incorporen condicionals, bucles, temps d'espera... i poden ser executats de forma autònoma, opció que pot ser útil en certes ocasions.

9.4 Escenes

Les escenes enregistren l'estat d'un grup de dispositius a un estat predefinit. Una escena és una llista de dispositius i l'estat que es desitja que estiguin. Quan una escena s'activa, els dispositius configurats en aquesta adquireixen el estat definit a l'escena.

Taula 12. Exemple de configuració de l'arxiu scenes.yaml

scenes.yaml	configuration.yaml
# Exemple escena - name: lectura entities: light.tauletanit: state: on brightness: 50 light.dormitori: state: "off"	*scene: !include scenes.yaml*

9.5 Interfície d'usuari

Com hem introduït abans, podem accedir a la interfície d'usuari amb http://direcció_ip:8123. Home Assistant tracta de facilitar la configuració de tot el sistema per mitjà d'aquesta interfície. Tot i que a l'actualitat no es pot configurar tot el conjunt amb la interfície, les característiques més usuals que hem anat desgranant fins aquí són configurables via web.

El primer que ens trobem quan accedim, després d'autentificar-nos, és el panell Lovelace amb totes les targetes configurades. Per defecte, gracies al sistema de detecció automàtica, Home Assistant incorpora algunes targetes de forma inicial.

La resta d'opcions les trobem al menú lateral. Podem destacar un registre seqüencial dels canvis d'estats dels nostres dispositius. A l'historial podem consultar d'un colp d'ull tots els estats dels dispositius i canvis en un tram horari a la nostra elecció. I finalment, al apartat de configuració, tenim accés als apartats de configuració del sistema i dels dispositius, personalització, automatitzacions i guions.

10 Rhasspy

Rhasspy és un conjunt de ferramentes amb una única interfície web per gestionar-les i la comunicació entre els components es realitza amb MQTT . D'aquesta manera, l'usuari final té la facilitat de seleccionar les aplicacions segons les seves necessitats i construir un assistent personal complet.

El primer sistema necessari és l'enregistrament d'àudio. Es pot enregistrar amb un micròfon local o per mitja de un flux d'àudio remot. L'assistent té disponibles Alsa, PyAduio i MQTT/Hermes. Alsa és un component del nucli Linux amb el fi de configurar automàticament les targetes de so tant domestiques com professionals. PyAudio proporciona enllaços Python per PortAudio, que és una biblioteca de codi obert, per a la reproducció i gravació d'àudio. Per últim, pot rebre àudio mitjançant MQTT amb format *wav*.

El següent sistema és el reproductor de so, que serà l'encarregat de reproduir la sortida de veu quan sigui necessària. Entre les opcions disponibles, tenim, com a l'enregistrament de veu, de MQTT/Hermes i Alsa. Rhasspy també pot enviar dades d'àudio a un dispositiu remot via HTTP.

El sistema w*ake word* permet activar l'enregistrador de veu a traves d'una paraula clau. D'aquesta forma, una volta el sistema detecte la paraula vigília, s'activarà el registrament de veu, mentrestant, no es registrarà cap cosa. Aquesta funció és molt habitual als dispositius actuals com Alexa o Siri. Destaquem Pocketsphinx, el sistema més flexible dels disponibles, que permet definir qualsevol paraula com a vigília, però recomanen paraules amb 3-4 síl·labes. Tot i tenir un pitjor rendiment en falsos negatius o positius, ofereix l'opció de configurar el llindar per ajustar-se a la paraula clau i millorar el rendiment. Porcupine en canvi, malgrat oferir millor rendiment, és més estàtic a l'hora de definir la paraula clau. Per defecte utilitza la paraula 'porcupine' i té disponibles d'altres a un repositori al web. Si s'opta per una paraula personalitzada, sols és vàlid durant 30 dies, i passat aquest temps has de tornar a generar-se l'arxiu.

Una de les funcions principals de Rhasspy és la transcripció d'ordres de veu a text. Aquesta funció es denomina STT i també s'encarrega de detectar els límits d'una comanda de veu, és a dir, quan una persona comença a parlar i quan acaba. De tots els sistemes disponibles només tenim un que admet l'idioma català i espanyol i ho fa sense connexió a Internet. Es tracta de Pocketsphinx, que ja hem comentat a la secció de '*wake work*'. Aquest sistema es restringeix al diccionari, model de llenguatge i model acústic que s'han utilitzat durant l'entrenament de Rhasspy.

El sistema de reconeixement d'intencions s'encarrega del text resultant del STT i d'identificar la intenció a la qual pertany. Un *intent* és un identificador per executar una tasca determinada. Fsticuffs és menys flexible però molt ràpid tant en l'entrenament com el reconeixement. Rhasspy recomana aquesta opció si sols es té previst el reconeixement de comandes a traves de veu i no de xat de text. Fuzzywuzzy troba la intenció més propera mitjançant la distància Levenshtein entre el text i totes les frases d'entrenament enregistrades. Ofereix un millor rendiment amb un nombre reduït de frases i té certa resistència a errors ortogràfics.

Després de la transcripció de l'ordre de veu i reconèixer correctament la intenció, Rhasspy ja pot enviar un esdeveniment JSON a un altre sistema com Home Assistant. Per a la configuració amb Home Assistant és necessari crear un testimoni d'accés al controlador.

Per últim, una volta s'ha gestionat la comanda de veu, pot generar-se una sortida de veu com a resposta a la comanda. Per aconseguir-ho es necessita un sistema de conversió de text a veu o TTS. El sistema per defecte és eSpeak, un sintetitzador de la parla de codi obert però que sona robòtic. A més a més, és l'únic que té disponible català i no requereix connexió externa.

11 Configuració del sistema

11.1 Xifratge amb SSL/TLS

El primer que realitzarem serà encriptar les comunicacions amb les interfícies. Per xifrar les comunicacions entre els nostres dispositius i el programari utilitzarem OpenSSL, un paquet de ferramentes i biblioteques de codi obert que implementa funcions criptogràfiques dels protocols SSL i TLS. Per a la nostra implementació utilitzarem un certificat auto-signat, és a dir, que no està signat per una autoritat de certificació (CA). Llavors, aquests certificats són més fàcils de crear i no tenen cap cost econòmic. Això no obstant, com els navegadors no poden verificar la CA és molt probable que apareguin avisos de seguretat del certificat. Però, més enllà de la molèstia del avís al navegador, el nostre sistema oferirà un grau més de seguretat.

Per crear un certificat necessitem tenir instal·lat OpenSSL.

```
sudo apt install openssl
```

I amb aquest única comanda es crearà el certificat i la clau, ambdós necessaris per a la encriptació. Utilitzem un algorisme de xifratge de clau pública de 4096, mida recomanada a partir de 2020, i un format -x509, l'estàndard del format del certificats de clau pública que estan auto-signats.

```
openssl req -sha256 -newkey rsa:4096 -nodes -keyout privkey.pem -x509 -days
730 -out fullchain.pem
```

Després d'executar aquesta comanda tindrem dos arxius, *fullchain.pem* i *privkey.pem*, els quals utilitzarem a tot el programari que volem xifrar les comunicacions. Per afegir el certificat a les nostres interfícies cal modificar diversos arxius de configuració i crear les rutes necessàries amb el certificat i la clau provada.

- A Home Assistant afegim la entrada *http* a la configuració, i indiquem les rutes del arxius generats:

http:

ssl_certificate: /ssl/fullchain.pem

ssl_key: /ssl/privkey.pem

- A Rhasspy, cal modificar la comanda d'arrancada i afegir:

  --certfile /profiles/ca/ssl/fullchain.pem --keyfile /profiles/ca/ssl/privkey.pem

- I finalment a Portainer afegim a la comanda d'arrancada:

  /home/pi/Docker/Portainer/local-certs:/certs --ssl --sslcert /certs/fullchain.pem --sslkey / certs/privkey.pem

Portainer facilita la tasca de modificar els paràmetres d'arrancada amb la seva interfície i sense el inconvenient de buscar els arxius de configuració dels contenidors.

Arribat a aquest punt el nostre programari iniciarà amb el xifratge. Un cop configurem l'excepció al nostre navegador, ja podrem utilitzar qualsevol interfície del programari.

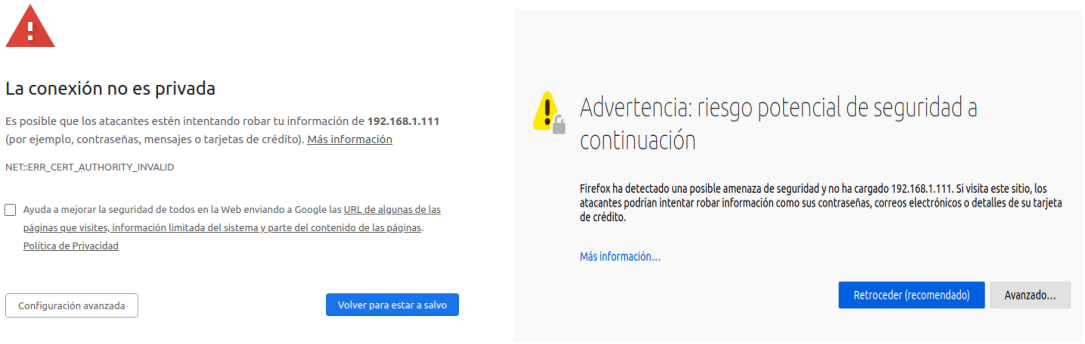

Figura 4. Avisos del certificat auto-signat als navegadors Firefox i Chrome

Els certificats auto-signats tenen com a inconvenient que les aplicacions d'Android no els admeten. Per obtenir certificats vàlids, hem de tenir un DNS públic. Un servei molt emprat per aquestes configuracions, són els DNS dinàmics com DuckDNS i l'autoritat certificadora Let's Encrypt. Però, per als propòsits d'aquest treball serà suficient amb aquests certificats.

11.2 Configuració inicial de Home Assistant

Tot i que després de la instal·lació vam configurar les variables bàsiques amb la interfície web, afegirem les dades manualment a l'arxiu *configuration.yaml* de Home Assistant. Els següents són una mostra de paràmetres disponibles:

- *latitude*: latitud de la teva localització.
- *longitude*: longitud de la teva localització.
- *elevation*: metres sobre el nivell del mar.
- *unit_system*: admet dos valors, *metric* per al sistema mètric i *imperial* per al sistema imperial. També és possible utilitzar un sistema de temperatura diferent amb el paràmetre *temperature_unit* amb valor *C* per a Celsius i *F* per a Fahrenheit.
- *time_zone*: zona horària.
- *name*: nom de la localització de Home Assistant.

Després d'haver creat les entrades corresponents amb les dades de geolocalització a *secrets.yaml*, el nostre arxiu quedarà així:

```
Homeassistant:
  latitude: !secret home_latitude
  longitude: !secret home_longitude
  elevation: 50
  unit_system: metric
  time_zone: Europe/Madrid
  name: Dani TFG
```

Cal informar que Home Assistant incorpora de manera automàtica la integració amb Meteorologisk Institutt de la qual rep la informació meteorològica basant-se amb les dades d'ubicació proporcionades al programari i que són accessibles amb el sensor *weather*. I també ofereix informació sobre la ubicació del Sol, molt útil per a les automatitzacions a l'alba i al vespre, disponible a la identitat *sun*. Les dues les emprarem per oferir funcionalitats al sistema.

11.3 Seguiment de dispositius

Amb el seguiment de dispositius es detecta quines persones són a una localització concreta amb la informació rebuda dels dispositius connectats. Com no anem a fer servir Home Assistant fora de la nostra xarxa, únicament utilitzarem aquesta funcionalitat per detectar la presencia de persones al domicili.

A la nostra implementació farem servir el Bluetooth, que comprovarà periòdicament els dispositius connectats. Per habilitar el seguiment, hem d'instal·lar els següents paquets a la nostra Raspberry pi.

```
sudo apt install bluetooth libbluetooth-dev
```

Inclourem el rastrejador, *device_tracker*, a l'arxiu de configuració amb un interval de 60 segons entre cada exploració. Amb el paràmetre *track_new_device*s amb valor *true* enregistrarem els nous dispositius. Els dispositius que es detecten amb el rastrejadors com Bluetooth es poden trobar a un arxiu que es crearà de forma automàtica al directori de configuració de Home Assistant amb l'identificador *known_devices.yaml. S*i tenim detectats els dispositius els quals volem fer el seguiment, podem desactivar el seguiment de la resta de aparells posant a *false* el paràmetre *track_new_device*. Desactivar el rastreig de nous dispositius evita el seguiment dels aparells, però es seguirà guardant la informació bàsica dels dispositius Bluetooth que es detecten a l'arxiu *known_devices.yaml.*

Mostrem un exemple de dispositiu trobat amb el rastrejador Bluetooth i que podem modificar si ho trobem necessari.

```
dani_rock:
  name: portàtil
  mac: BT_XX:XX:XX:XX:XX:XX
  icon:
  picture:
  track: true
```

Amb el paràmetre *track* amb el valor *true* indiquem quins dispositius volem fer seguiment. Per contra, amb el valor *false*, no es realitzarà el seguiment.

A continuació crearem la zona que utilitzarem per assignar la ubicació en la qual es troba l'usuari. Aquesta zona l'anomenarem 'Home' i ens permetrà utilitzar-la com activador o condició en una automatització. També afegirem una entitat de tipus persona amb el nom 'Daniel'.

I finalment, assignarem dos dispositius a la persona, un mòbil i un ordinador portàtil. Per tant, si el rastrejador detecta algun d'aquests dispositius assignarà a la entitat de la persona a la zona creada anteriorment.

Aquesta configuració es pot veure a l'arxiu *configuration.yaml* a l'Annex: Arxius de Home Assistant, a les seccions *zone* i *person*.

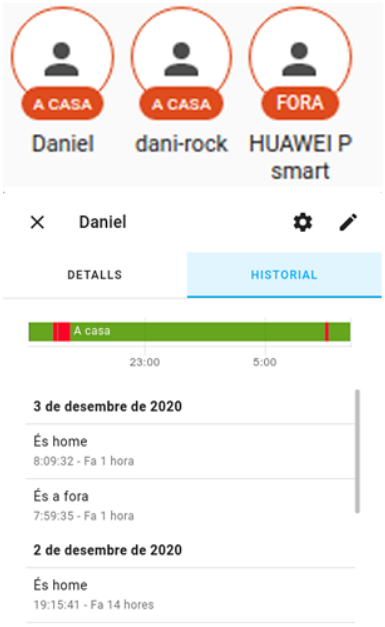

Figura 5. Home Assistant. Rastreig de persones amb dispositius

11.4 Afegir dispositius ZigBee

El nostre xip CC2531, connectat a un port USB de la nostra Raspberry, serà el pont entre la xarxa ZigBee i Home Assistant. Per configurar el programari hem de dirigir-nos a la interfície web a la secció d'integracions, que es troba dintre del apartat de configuració. Home Assistant ofereix la integració anomenada ZHA, per facilitar la tasca de configuració de nous dispositius. Cal seleccionar la porta de enllaç, al nostre cas el xip CC2531 de Texas Instruments, a la configuració. Després d'uns pocs segons, la porta d'enllaç ja estarà agregada correctament.

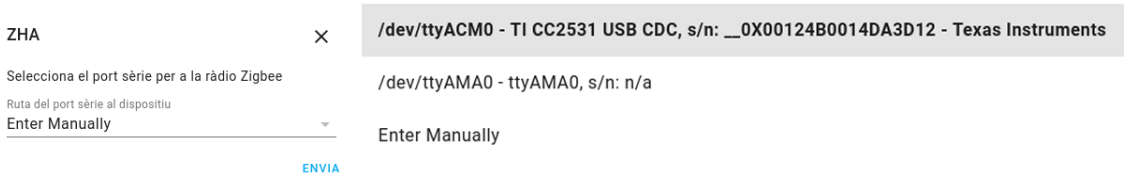

Figura 6. Finestres de selecció del coordinador ZigBee

La resta de dispositius ZigBee s'agregaran a través d'aquest dispositiu. Per sincronitzar els dispositius amb el coordinador hem d'habilitar l'emparellament als aparells desitjats. Segons el dispositiu i el fabricant les instruccions poden variar. Els nostres dispositius s'activen d'aquesta forma:

- Bombeta Ikea: apagar i encendre amb el interruptor 6 voltes.
- Endoll Ikea: disposa de un botó que ha de polsar-se fins que un led parpelleja.
- Sensor Temperatura/Humitat Sonoff: polsar el botó que disposa fins que s'encén un led.

Figura 7. Finestra d'informació del coordinador ZigBee

Una volta finalitzar aquest procés, ja tenim disponibles tots els dispositius i entitats de la xarxa ZigBee al nostre sistema.

Figura 8. Dispositius trobats amb el coordinador ZigBee

11.5 Creació d'escenes

Es crearan dues escenes a Home Assistant, les quals modificaran la intensitat de llum del despatx. Una tindrà la brillantor al 100% que s'utilitzarà per treballar i una altra amb el brillantor al 70% que anomenarem lectura. Mostrem com a exemple l'escena 'treball'.

scenes.yaml

```
- name: treball
  entities:
    light.llum_despatx:
      state: true
      brightness: 254
```

Figura 9. Escenes a Home Assistant

11.6 Creació d'automatitzacions

A aquest apartat es crearan les automatitzacions de Home Assistant que dotaran d'un poc d'intel·ligència al sistema. Les automatitzacions, continuant amb la separació d'arxius per facilitar el manteniment, han d'escriure's a *automations.yaml.*

La primera automatització tracta d'oferir un estalvi de consum de llum a la llar. Per això, connectarem el escalfador d'aigua al endoll ZigBee que es connectarà únicament a la franja horària nocturna, és a dir de 0h a 8h.

```
- id: escalfador_on
  alias: escalfador_on
  description: Encendre escalfador d'aigua a
mitjanit
  trigger:
  - platform: time
    at: 00:00
  action:
  - service: homeassistant.turn_on
    entity_id: switch.endoll
  mode: single
```

```
- id: escalfador_off
  alias: escalfador_off
  description: Apagar escalfador d'aigua al matí
  trigger:
  - platform: time
    at: 08:00
  action:
  - service: homeassistant.turn_off
    entity_id: switch.endoll
  mode: single
```

La següent automatització encendrà el llum del despatx al vespre quan estigui l'ordinador encès. Per encendre el llum farà ús de la escena treball, i es llançarà 25 minuts abans del vespre.

```
- id: es_fa_de_nit
  alias: es_fa_de_nit
  mode: single
  description: encendre els llums al despatx quan es fa de nit
  trigger:
  - platform: sun
    event: sunset
    offset: -00:25:00
  condition:
  - condition: time
    weekday:
    - mon
    - tue
    - wed
    - thu
    - fri
  - condition: state
    state: home
    entity_id: device_tracker.dani_rock
  action:
  - scene: scene.treball
```

11.7 Sensor d'hora

Perquè el sistema pugui dir-nos l'hora actual, hem d'afegir un sensor d'aquest tipus a l'arxiu de configuració principal de Home Assistant. Aquest sensors incorporen l'actualització cada minut i permet produir dades amb un format personalitzat.

```
Sensor:
  - platform: time_date
    display_options:
      - 'time'
```

11.8 Connexió de Rhasspy amb Home Assistant

Els testimonis d'accés s'utilitzen per a la integració de programari aliè i ens servirà per executar comandes des de Rhasspy. Aquests testimonis són vàlids durant 10 anys i podem crear-los amb la interfície web. A la secció anomenada 'Tokens d'autenticació d'accés de llarga durada' del nostre perfil, crearem un testimoni amb el nom 'Rhasspy'.

Figures 10 i 11. Testimonis d'accés de Home Assistant

El valor d'aquest testimoni d'accés hem d'afegir-lo a l'arxiu de configuració del nostre perfil de Rhasspy, junt amb l'adreça de Home Assistant.

Secció de l'arxiu profile.json

```
"home_assistant": {
  "access_token":  "eyJ[...]MDU",
  "url":           "https://tfg:8123"
},
```

11.9 Intencions

Les intencions són una descripció d'allò que un usuari desitja que és faci. Al nostre escenari, Rhasspy convertirà una comanda de veu a una intenció que serà traslladada a Home Assistant per gestionar-la. Per testejar el bon funcionament, cada intenció retornarà a Rhasspy un text que serà convertit a

veu. Tant Rhasspy com Home Assistant han d'utilitzar els mateixos identificadors de les intencions per a poder comunicar-se. A més de definir les nostres pròpies intencions, Home Assistant inclou unes poques de bàsiques amb funcionalitats similars a les que ofereixen els serveis del sistema. A Home Assistant no cal afegir cap codi, ja que automàticament es cridaran a aquests serveis. En canvi, per fer ús d'aquestes funcionalitats bàsiques hem d'afegir la entrada corresponent a les sentències de Rhasspy.

Cal destacar les intencions *HassTurnOn, HassTurnOff* i *HassToogle*, per a encendre, apagar o commutar l'estat del dispositiu.

```
[HassTurnOn]
    Encén el (llum del despatx){name}
    Encén l'(endoll){name}
[HassTurnOff]
    Apaga el (llum del despatx){name}
    Apaga l'(endoll){name}
```

O també tenim la opció d'utilitzar *HasLightSet* per a les opcions de color i brillantor dels llums.

```
[HassLightSet]
    Posa [el] (llum del despatx){name} al (màxim:100 | mínim:10 | mig:50)    {brightness}
```

Rhasspy admet l'ús de plantilles a les intencions: les claus són utilitzades per enviar l'identificador de la identitat a Home Assistant, els parèntesis agrupen diverses paraules, els dos punts són emprats per a substitucions de paraules i els claudàtors indiquen paraules opcionals. També es poden crear variables i cridar-les amb '<*nom_variable*>' o llistes de ranures, emmagatzemades al directori *slots* a un arxiu sense extensió, que poden referenciar-se amb el símbol $.

Home Assistant també admet plantilles als seus arxius. Aquestes estan basades amb Jinja2, un motor de plantilles per a Python que admet estructures de control, operacions matemàtiques i comparacions.

Com aquestes primeres intencions són molt bàsiques i no podem configurar tampoc el text de retorn cap a Rhasspy, necessitem definir unes de pròpies tant a Home Assistant com a Rhasspy. Al controlador domèstic utilitzarem l'arxiu *intent_script.yaml* i a l'assistent de veu *sentences.ini*, com per exemple:

sentences.ini

```
[ChangeLightStateDespatx]
states = (Encén | Apaga ) {light_state}
<states> el despatx
```

intent_script.yaml

```
ChangeLightStateDespatx:
 action:
 - service_template: >
     {%- if light_state == 'Encén' -%}
       homeassistant.turn_on
     {%- else -%}
       homeassistant.turn_off
     {%- endif -%}
   data_template:
     entity_id: light.llum_despatx
 speech:
   text: >
     Fet. {{ light_state }} el despatx
```

Afegim la funcionalitat que retorna l'hora actual.

sentences.ini

```
[GetTime]
  quina hora és
  dime l'hora
```

intent_script.yaml

```
GetTime:
 speech:
   text: Ara són les {{ states.sensor.time.state }}
```

Amb el mateix procediment, implementarem *GetTemperatureInt* i *GetTemperatureExt* que respondran a les comandes 'quina és la temperatura a casa' i 'quina és la temperatura afora', i la comanda per activar les escenes, *ActiveScene.*

Per últim, crearem les intencions *Predicció* i *PrediccioDia,* que utilitzarà un script de Python a Home Assistant. La creació del guió l'exposarà automàticament com a servei, el qual serà cridat des de les intencions. Amb aquest guió recopilarem tota la informació de la predicció del temps de ens ofereix el sensor *weather* i la manipularem per oferir les dades a Rhasspy. Per emmagatzemar les dades del guió necessitem crear un sensor nou, '*predicció*', i habilitar *python_script* a la configuració de Home Assistant. D'aquesta forma, el script ja pot ser cridat com a un servei.

intent_script.yaml

```
Prediccio:
  action:
    - service: python_script.prediccio
  speech:
    text: >
     {%- if states.sensor.prediccio == "no hi ha dades" -%} No hi ha dades
      {%- else -%} {{ states.sensor.prediccio.attributes }}
      {%- endif -%}
```

Tan el codi Python creat, *prediccio.py,* com la resta d'arxius pot ser consultat a l'Annex: Arxius de Home Assistant i l'Annex: Arxius de Rhasspy.

11.10 Proves de funcionament d'intencions

Amb tot el sistema configurat i creades totes les intencions que farem servir, posarem a prova la funcionalitat del sistema. Rhasspy utilitza l'arxiu */var/log/messages* per enregistrar el diari d'esdeveniments. El modul encarregat de gestionar les intencions és Rhasspy Hermes Dialogue Manager. Per tant, farem un seguiment del diari amb la següent comanda:

```
tail -f /var/log/messages | cut -d " " -f 8- | grep rhasspyhomeassistant_hermes
```

Executem la comanda per saber l'hora, i com podem apreciar, davant la entrada de veu 'quina hora és', detecta que la intenció és *GetTime*, que envia a la API de Home Assistant i rep el missatge de veu 'Ara són les 12:52'

*rhasspyhomeassistant_hermes: <- NluIntent(**input='quina hora és',***
*intent=Intent(i**ntent_name='GetTime'**, confidence_score=1.0), [...]*
*rhasspyhomeassistant_hermes: **https://tfg/api/intent/handle***
*rhasspyhomeassistant_hermes: -> **TtsSay(text='Ara són les 12:52',** site_id='default',*
lang=None, id='0f796034-a945-4f08-bf1c-f310f7c2f322', session_id='1e1815b5-79f8-4626-b9de-2b4465fc9edd', volume=None)

Després de provar *GetTemperatureInt* i *GetTemperatureExt o*btenim resultats idèntics. Ambdós han funcionat també correctament.

A continuació utilitzarem la intenció *ChangeLightStateDespatx,* que modificava l'estat d'una entitat de Home Assistant per comprovar el funcionament amb enviament de paràmetres.

*rhasspyhomeassistant_hermes: <- **NluIntent(input='Encén el despatx',***
intent=Intent(intent_name='ChangeLightStateDespatx', confidence_score=1.0)
rhasspyhomeassistant_hermes: https://tfg/api/intent/handle
*rhasspyhomeassistant_hermes: -> **TtsSay(text='Fet. Encén despatx',** site_id='default',*
lang=None, id='4ba1a79f-2ac3-4010-bc29-22f86dc043f8', session_id='default-escolta-310a1c2d-d52f-4f30-9061-12fa76d0a787', volume=None)

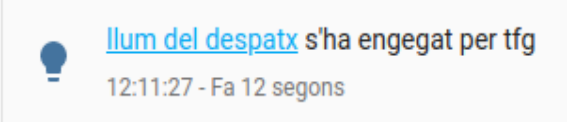

Figura 12. Canvi d'estat al registre de Home Assistant

Ara utilitzarem la intenció *ActiveScene,* que activa l'escena enviada com a paràmetre.

rhasspyhomeassistant_hermes: <- ***NluIntent(input='activa escena de treball',*** *intent=Intent(intent_name='ActiveScene', confidence_score=1.0))*

rhasspyhomeassistant_hermes: <- ***NluIntent(input='activa escena de lectura',*** *intent=Intent(intent_name='ActiveScene', confidence_score=1.0))*

En aquest cas, si consultem el diari de Home Assistant que trobarem al directori de configuració baix el nom *home-assistant.log, p*odem veure com s'activa l'escena durant l'execució de la intenció.

[homeassistant.helpers.script.intent_script_activescene] ***Intent Script ActiveScene: Running intent_script script***
[homeassistant.helpers.script.intent_script_activescene] Intent Script ActiveScene: Executing step call service
[homeassistant.core] Bus:Handling <Event call_service[L]: ***domain=scene, service=turn_on, service_data=entity_id=scene.treball****>*
[homeassistant.helpers.script.intent_script_activescene] ***Intent Script ActiveScene: Running intent_script script***
[homeassistant.helpers.script.intent_script_activescene] Intent Script ActiveScene: Executing step call service
[homeassistant.core] Bus:Handling <Event call_service[L]: ***domain=scene, service=turn_on, service_data=entity_id=scene.lectur****a>*

I per últim utilitzarem les intencions *Prediccio* i *PrediccióDia* que executen el guió de Python i rep la predicció dels pròxims dies o del dia indicat.

rhasspyhomeassistant_hermes: <- ***NluIntent(input='Quin temps farà diumenge',*** *intent=Intent(intent_name='PrediccioDia', confidence_score=1.0)*
rhasspyhomeassistant_hermes: https://tfg/api/intent/handle
rhasspyhomeassistant_hermes: -> ***TtsSay(text=' estarà parcialment ennuvolat amb una temperatura màxima de 18.1. .'****, site_id='default', lang=None, id='d94a3e39-b61b-4a59-9b66-f6dce5c78ac9', session_id='default-escolta-7a3c0a91-978b-4e1e-a086-f0a8910b1a00', volume=None)*

12 Conclusions

Aquest projecte tenia la finalitat d'oferir un sistema domòtic amb assistent de veu que inclogués diversos dispositius. Aquest devia implementar-se amb un sistema GNU/Linux i calia que fora de codi obert per poder mantenir controlades i protegides la informació i les dades personals. Per aquest mateix motiu, cap element del programari devia d'enviar dades fora de la nostra xarxa.

Hem realitzat un estudi dels protocols de comunicació més utilitzats en els darrers anys i hem posat especial atenció als mecanismes de seguretat que oferien: autentificació, integritat, confidencialitat i frescor. Tot seguit, hem realitzat un anàlisi de les diferents opcions tant de programari com de maquinari que complien amb els requisits d'accés al codi i oferiren mecanismes de seguretat.

Finalment, hem desplegat el programari Home Assistant i Rhasspy amb una Raspberry Pi. Els dispositius de diversos fabricants, tant sensors com actuadors, es comuniquen amb el protocol ZigBee mitjançat un coordinador implementat amb un xip CC2531 amb el microprogramari Z-Satck.

D'una banda, al controlador domòtic hem tractat d'oferir funcions de gestió d'energia i automatitzacions. Per l'altre costat, hem connectat l'assistent de veu amb el controlador domèstic per realitzar tasques mitjançant la veu, utilitzant el català com a llengua.

Tot aquest sistema ha estat desplegat amb mecanismes de seguretat, ja siguin els propis del protocol ZigBee, o mitjançant l'autentificació amb les interfícies web. Encara més, hem utilitzat SSL/TLS per a protegir les comunicacions amb les interfícies i implementat mecanismes de control bàsic al sistema operatiu.

Arribat a aquest punt, considerem assolits tots els objectius descrits.

13 Glossari

UNIX: sistema operatiu multiusuari i multitasca que va ser desenvolupat a finals de la dècada dels 60 per un grup d'empleats dels laboratoris d'AT&T Bell.

Entrada/sortida (**E/S**): intercanvis d'informacions entre el processador i els perifèrics que li són associats en un sistema basat en un processador.

GNU Compiler Collection (GCC): és un sistema compilador de llenguatges de programació realitzats pel projecte GNU. El GCC el va escriure originàriament Richard Stallman l'any 1987 per usar-lo de compilador de C en el GNU i oferir-lo gratuïtament com a programari lliure. Posteriorment s'hi van afegir altres llenguatges. Ha estat adoptat com compilador oficial utilitzat per construir i desenvolupar el sistema operatiu GNU, i com a estàndard en la majoria de distribucions Linux.

Linux Foundation: consorci tecnològic sense ànim de lucre fundat el 2000 amb la finalitat de estandarditzar el sistema Linux, donar suport i promoure la seva adopció.

Tunelització (*tunneling*): tècnica que consisteix a encapsular un protocol de xarxa sobre un altre creant un túnel dins d'una xarxa d'ordinadors.

Encaminament (routing): és la funció de buscar un camí entre tots els possibles en una xarxa de paquets les topologies dels quals posseeixen una gran connectivitat.

IDE (*Integrated development environment*): és un entorn de desenvolupament que agrupa diferents funcions en un sol programa, habitualment: editor de codi, compilador, depurador i un programa de disseny d'interfície gràfica.

Blocky: biblioteca de JavaScript amb l'objectiu de crear blocs de codi de manera visual, on l'usuari pot seleccionar una serie de blocs disponibles i arrastrar-lo a un tauler de treball. Normalment s'executa en un navegador.

OSGi: és un entorn de treball de Java per desenvolupar i desplegar mòduls de programari de manera dinàmica.

IoT: Les sigles de Internet de les coses (*Internet of Things*) es refereix a una xarxa d'objectes de la vida quotidiana interconnectats.

Copyleft: pràctica legal que consisteix en l'exercici del dret d'autor amb l'objectiu de propiciar el lliure ús i distribució d'una obra, exigint que els concessionaris preservin les mateixes llibertats al distribuir les seves còpies i derivats.

Llicència Apache: llicència de programari lliure creada per Apache Software Foundation que requereix la conservació de l'avís de dret d'autor però no requereix la redistribució del codi font a les versions modificades.

WebSocket: tecnologia que proporciona un canal de comunicació bidireccional sobre un únic sòcol TCP.

Buildroot: ferramenta de codi obert per automatitzar la creació de distribucions basades en Linux per a maquinari encastat.

SSL: *Secure Sockets Layer,* un protocol criptogràfic.

TLS: *Transport Layer Security*, un protocol criptogràfic.

SoC: és un circuit integrat, xip, que integra totes o la majoria de components d'un ordinador o d'un altre sistema electrònic.

Transceptor: Un transceptor és un dispositiu que compta amb un transmissor i un receptor que comparteixen part del circuit elèctric i es troben dins de la mateixa caixa o placa.

Kubernetes: és un sistema de codi obert per implementar i desplegar aplicacions distribuïdes.

WAV: és un format d'àudio digital desenvolupat per IBM i Microsoft, normalment sense compressió de dades.

Distància Levenshtein: és el nombre mínim de operacions necessàries per transformar una cadena de caràcters a una altra.

DuckDNS: servei gratuït que dirigeix un DNS a una IP dinàmica.

Let's Encrypt: autoritat de certificació gratuita.

14 Bibliografia

14.1 Llibres – Manuals

Free Software, Free Society.(2002). Richard M. Stallman. ISBN 978-0-9831592-0-9

KNX Conocimientos básicos.
https://www.knx.org/wAssets/docs/downloads/Marketing/Flyers/KNX-Basics/KNX-Basics_es.pdf
[consultat: 14/10/2020]

2020 Linux Kernel History Report (2020). Linux Foundation.
https://www.linuxfoundation.org/wp-content/uploads/2020/08/2020_kernel_history_report_082620v2.pdf [consultat: 27/10/2020]

Programming Voice Interfaces (2017). Walter Quesada, Bob Lautenbach. ISBN:
9781491956069 https://www.oreilly.com/library/view/programming-voice-interfaces/9781491956052/

Documentation - Mycroft AI https://mycroft-ai.gitbook.io/docs/ [consultat: 5/10/2020]

Rhasspy Voice Assistant https://rhasspy.readthedocs.io/ [consultat: 9/10/2020]

Knx Data Secure
https://download.gira.de/data3/KNX_Data_Secure_Systemdokumentation_en.pdf [consultat: 24/10/2020]

Make Magazine Volume 72 Spring 2020 [consultat 10/11/2020]

14.2 Articles

Why is open source software more secure?(2015). Jim Lynch.
https://www.infoworld.com/article/2985242/why-is-open-source-software-more-secure.html
[consultat: 26/10/2020]

Manifiesto en favor de la transparencia en desarrollos de software públicos
https://transparenciagov2020.github.io/ [consultat: 27/10/2020]

Home automation.
https://en.wikipedia.org/wiki/Home_automation [consultat: 27/10/2020]

Domòtica
https://en.wikipedia.org/wiki/Home_automation [consultat: 27/10/2020]

Z-Wave is making a huge change so it doesn't get left behind in the smart home wars (2019). Jacob Kastrenakes. https://www.theverge.com/2019/12/19/21029661/zwave-open-standard-radios-smart-home-multiple-vendors-silicon-labs [consultat: 16/10/2020]

Apple, Google, and Amazon are teaming up to develop an open-source smart home standard (2019). Jacob Kastrenakes. https://www.theverge.com/2019/12/18/21027890/apple-google-amazon-smart-home-standard-zigbee-connected-ip-project [consultat: 16/10/2020]

El estándar de 'smart home' respaldado por Google, Apple y Amazon se lanzará en 2021 (2020). Portaltic/EP. https://www.europapress.es/portaltic/sector/noticia-estandar-smart-home-respaldado-google-apple-amazon-lanzara-2021-20200909145640.html [consultat: 16/10/2020]

El sistema KNX al detalle. https://partner.gira.com/es_ES/gebaeudetechnik/systeme/knx-eib_system/systemerlaeuterung.html [consultat: 14/10/2020]

KNX. https://es.wikipedia.org/wiki/KNX [consultat: 14/10/2020]

LonWorks. https://en.wikipedia.org/wiki/LonWorks [consultat: 15/10/2020]

X10 (industry standard). https://en.wikipedia.org/wiki/X10_(industry_standard) [consultat: 15/10/2020]

Zigbee. https://en.wikipedia.org/wiki/Zigbee [consultat: 15/10/2020]

Conectividad ZigBee de Amazon Echo Plus (2019). Yúbal Fernández. https://www.xataka.com/basics/conectividad-zigbee-amazon-echo-plus-que-como-funciona-otros-dispositivos-compatibles [consultat: 15/10/2020]

Z-Wave. https://en.wikipedia.org/wiki/Z-Wave [consultat: 16/10/2020]

Z-Wave Plus. https://www.domoticalia.es/es/content/13-z-wave-plus [consultat: 16/10/2020]

MQTT. https://en.wikipedia.org/wiki/MQTT [consultat: 16/10/2020]

Arduino. https://en.wikipedia.org/wiki/Arduino [consultat: 9/10/2020]

Raspberry Pi.
https://en.wikipedia.org/wiki/Raspberry_Pi [consultat: 12/10/2020]

Tinker Board | Single-board Computer | ASUS España.
https://www.asus.com/es/Single-Board-Computer/Tinker-Board/overview/ [consultat: 13/10/2020]

Odroid.
https://en.wikipedia.org/wiki/ODROID [consultat: 1310/2020]

Xtend.
https://es.wikipedia.org/wiki/Xtend [consultat: 20/10/2020]

Many Roads, One Destination (2018).
https://mycroft.ai/blog/many-roads-one-destination/ [consultat: 5/11/2020]

Mycroft (software)
https://en.wikipedia.org/wiki/Mycroft_(software) [consultat: 6/11/2020]

Seguretat de la informació
https://ca.wikipedia.org/wiki/Seguretat_de_la_informaci%C3%B3 [consultat: 24/11/2020]

Docker https://en.wikipedia.org/wiki/Docker_(software) [consultat: 19/11/2020

Kubernetes https://en.wikipedia.org/wiki/Kubernetes [consultat: 19/11/2020]

14.3 Webs

GNU Operating System https://www.gnu.org/ [consultat: 26/10/2020]

TOP 500 https://www.top500.org [consultat: 27/10/2020]

KNX Association KNX Association https://www.knx.org/ [consultat: 14/10/2020]

LonMark http://www.lonmark.es [consultat: 15/10/2020]

LonMark https://www.lonmark.org/ [consultat: 1/510/2020]

MQTT – The Standard for IoT Messaging https://mqtt.org/ [consultat: 16/10/2020]

Domoticz https://www.domoticz.com/ [consultat: 19/10/2020]

OpenHAB https://www.openhab.org/ [consultat: 21/10/2020]

Home Assistant https://www.home-assistant.io/ [consultat: 21/10/2020]

PiHome - Smart Heating Control https://www.pihome.eu/ [consultat: 22/10/2020]

Jeedom https://www.jeedom.com/ [consultat: 22/10/2020]

Mozilla IoT WebThings https://iot.mozilla.org/ [consultat: 23/10/2020]

Almond https://almond.stanford.edu/ [consultat: 6/10/2020]

Stanford Open Virtual Assistant Lab https://oval.cs.stanford.edu/ [consultat: 6/10/2020]

Mycroft – The Open Source Privacy-Focused Voice Assistant https://mycroft.ai/ [consultat: 5/10/2020]

Rhasspy Voice Assistant https://github.com/rhasspy [consultat: 10/10/2020]

Zigbee2mqtt documentation www.zigbee2mqtt.io [consultat: 17/10/2020]

Z-Stack-firmware https://github.com/Koenkk/Z-Stack-firmware [consultat: 17/10/2020]

Raspberry Pi https://www.raspberrypi.org/ [consultat: 23/10/2020]

Docker – Empowering App Development for Developers https://www.docker.com/ [consultat: 23/10/2020]

Portainer. Simple Container Management https://www.portainer.io [consultat: 23/10/2020]