Homefort

SOMMARIO

1. SCOPO

HomeFort è un sistema di gestione del comfort all'interno di un'abitazione.

L'obiettivo preposto è stato quello di creare un sistema automatico di regolazione di alcuni aspetti del comfort casalingo, permettendo l'interazione con l'utente, in modo che questo possa impostare i parametri del comfort e gestire direttamente alcuni componenti.
L'utente finale del prodotto avrà, quindi la facoltà di accendere o spegnere le luci e impostarne la luminosità, potrà chiedere informazioni sulle condizioni interne della casa (luminosità, umidità, temperatura e movimento) e sulle condizioni meteo.
Per quanto riguarda la configurazione del sistema automatico, sarà possibile impostare gli orari quotidiani in cui le luci dovranno essere sempre accese e la temperatura ideale che si vorrà mantenere nell'abitazione.

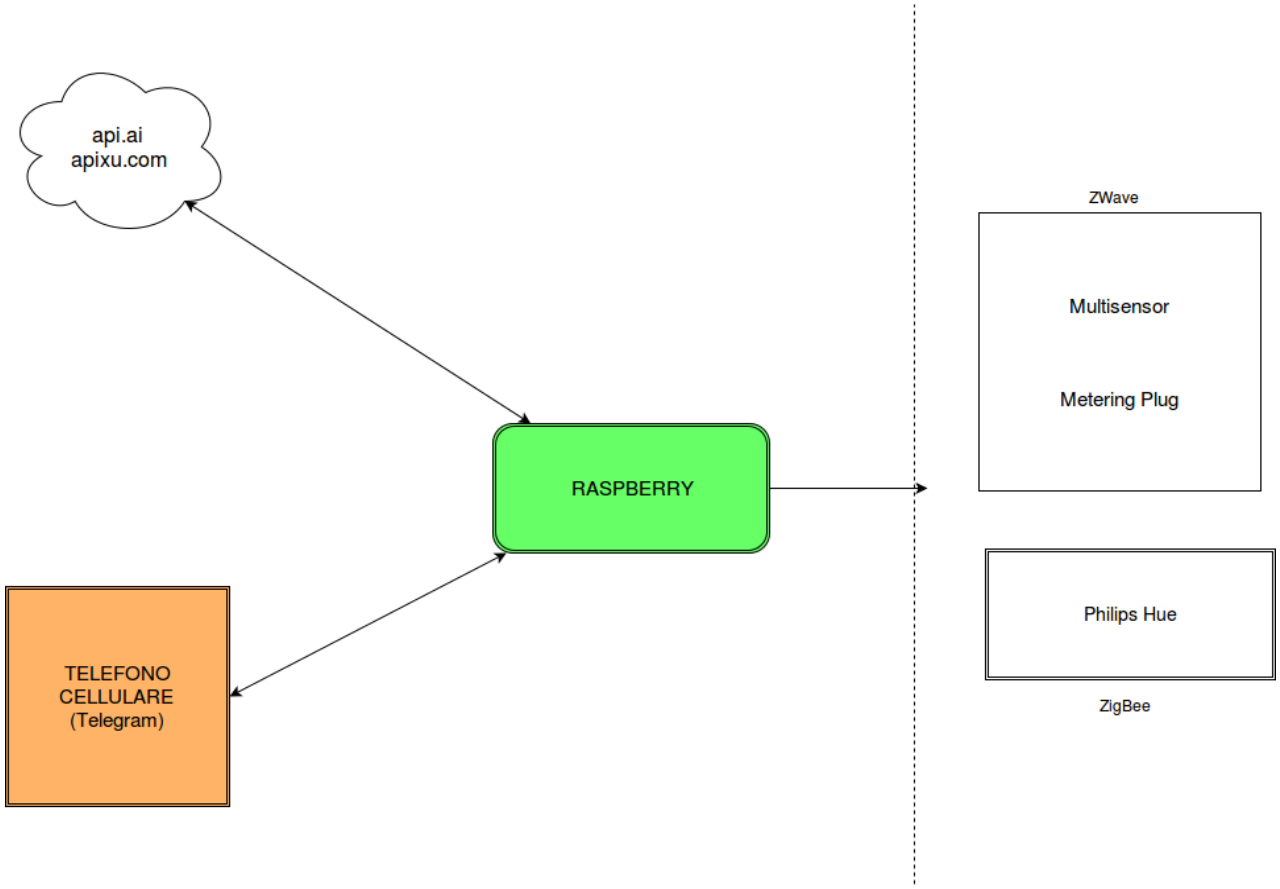
Il sistema di controllo automatico gestisce quindi luci e temperatura interna. Le prime verranno accese tutti i giorni nel range d'orario impostato in precedenzadall'utente, ma solamente se in casa verrà rilevato del movimento. HomeFort raccoglie informazioni sulla luminosità interna e imposta la luminosità delle luci intorno ad un valore ideale. Le due funzioni si attivano se richiesto dall'utente attraverso la modalità automatica.
La temperatura viene regolata automaticamente attorno al valore di comfort impostato da impostare.

L'interazione con l'utente è realizzata attraverso un assistente conversazionale che si appoggia su un bot Telegram, in modo che si possa gestire il sistema comodamente siafuori di casa, sia sul divano senza doversi muovere.

Nel dettaglio, i servizi forniti dal sistema sono:

- Accensione delle luci
- Impostazione della luminosità delle luci
- Spegnimento luci
- Impostazione di range orari per la gestione automatica delle luci
- Richiesta previsioni meteo su 3 giorni
- Richiesta per le condizioni meteo attuali
- Impostazione della temperatura ideale
- Possibilità di attivare/disattivare la gestione automatica delle luci (orario e luminosità.)
- Possibilità di controllare se qualcuno si trova nella casa.
- Richiedere le condizioni attuali della casa in termini di umidità, luminosità, temperatura e movimento.

2. ARCHITETTURA



Servizi Cloud di terze parti: si è scelto di usare il servizio API offerto da Apixu.com per la gestione delle previsioni meteo e il servizio di creazione di assistenti conversazionali e WebHook offerto da API.ai.

2.1. Api.ai

Attraverso l'interfaccia e le api di Api.ai è stata sviluppata tutta la parte che riguarda l'interazione con l'utente. Sono stati definiti i messaggi accettati, i default di avvio e di risposte non corrette, e infine la specifica del WebHook: a (quasi) ogni comando corrisponde un messaggio HTTP POST da parte di Api.ai rivolto indirettamente verso il nodo computazionale, contenente un oggetto JSON con la Action corrispondente al messaggio e altre informazioni. Le risposte del server sono dirette verso Api.ai (grazie alle sue api Java), che notifica l'utente.

I messaggi fra Api.ai ed il nodo computazionale avvengono indirettamente (anche) perché non è possibile contattare la rete scolastica dall'esterno. E' quindi è stato usato il servizio di ngrok: Api.ai imposta il WebHook su un URL esposto da questo software, che riceve i messaggi HTTP e li "memorizza"; poi un client installato sul nodo computazionale stesso si collega a quest'URL in modo da prelevare lui stesso i messaggi, invece di essere contattato dall'esterno.

Il servizio offerto da Api.ai permette inoltre l'integrazione automatica dell'assistente conversazionale con un bot Telegram.

Dispositivi Utente: l'utente può interagire col sistema tramite l'uso di un qualsiasi telefono cellulare connesso a internet, previa installazione dell'applicazione Telegram.

Nodo Computazionale: il programma che riceve e interpreta le richieste dell'utente e che gestisce il controllo del comfort è eseguito su Raspberry Pi.

Ambiente: sensori, luci e prese. Le luci sono Philips Hue e comunicano con un bridge comune tramite una rete ZigBee a protocollo proprietario e interrogabile tramite API REST. I sensori e le prese invece sono connesse ad una rete ZWave e comunicano con un gateway (modulo raZberry del Raspberry Pi).

2.2. Dettagli

Come si può osservare dalla struttura del progetto Java, si è scelto di dividere il sistema in 5 moduli principali che interagiscono tra loro (più una raccolta di funzioni di utilità). Il componente principale è quello denominato httpServer, che gestisce lo scambio di messaggi HTTP con le API di Api.ai e la parsificazione dei comandi in arrivo da Api.ai.

2.2.1. HttpServer

Il sotto-modulo **ApiAiListener** contiene il main che inizializza alcuni parametri (convertitore JSON, parametri ZWave), spegne tutti i dispositivi connessi alla rete (presa e Hue) e fa partire i threads necessari per controllare temperatura e luci. Dopo queste operazioni iniziali, aspetta continuamente l'arrivo di comandi sotto forma di messaggi HTTP: a questo scopo sono stati sfruttati i servizi offerti da Spark, un server leggero Java che, attraverso una funzione (post) attende messaggi HTTP POST e chiama un metodo specifico (doWebhook) non appena sono disponibili.

CommandExecuter implementa il metodo doWebhook appena citato: parsifica il comando appena ricevuto e delega alle varie funzioni disponibili il completamento dell'operazione richiesta; raccoglie i dati restituiti per costruire la risposta ed infine invia il messaggio HTTP .

I comandi disponibili prendono il nome dalle Actions dei rispettivi Intent definiti nella struttura dell'assistente conversazionale di Api.ai, presentano la stessa struttura dei comandi del bot Telegram (eccetto per '/' iniziale e per le maiuscole) e sono i seguenti:

- lightsOn: azione che permette all'utente di accendere autonomamente le luci. Esegue le operazioni necessarie all'accensione delle lampadine Philips Hue:
  - chiama il metodo corrispondente (lightsOn) dello hueController;
  - controlla se la modalità automatica è attiva, nel caso, la disattiva e scrive un messaggio per l'utente, avvisando della disattivazione della modalità automatica; poi accende la luce.
    La modalità automatica viene disattivata perché l'utente ha deciso di intervenire autonomamente sulla gestione delle luci: se volesse lasciare la gestione di nuovo nelle mani del sistema (che accende e spegne a seconda degli orari), allora dovrà riattivare la modalità automatica.

  Se qualcuna di queste operazioni fallisce, genera un'eccezione corredata da rispettivo messaggio d'errore. In ogni caso, alla fine dell'azione verrà inviato un messaggio all'utente.

- lightsOff: azione che permette all'utente di spegnere autonomamente le luci. Il funzionamento è sostanzialmente identico a quello dell'azione sopra descritta, l'unica differenza si trova nel metodo chiamato (lightsOff). Anche in questo caso viene disattivata la modalità automatica, se attiva.

- lightsPower: permette all'utente di impostare la luminosità delle luci. Il funzionamento, ancora una volta, è molto simile ai precedenti, le differenze si possono vedere nel metodo chiamato (lightsPower)e nella gestione del valore di percentuale dato in input dall'utente, il quale dovrà essere fra 0 e 100.

- time: permette di chiedere data e ora del giorno. Quest'azione richiede, semplicemente, l'ora di sistema del Raspberry e lo ritorna all'utente

- luminosity: riceve informazioni riguardanti la luminosità dell'ambiente.Richiama il metodo (getLuminosity) e ritorna il valore in Lux della luminosità.

- humidity: riceve informazioni sull'umidità interna della casa. Richiama il metodo (getHumidity) e ritorna il valore in % d'umidità.

- motion: informazioni sul movimento all'interno dell'abitazione. Richiama il metodo (getMotion) e ritorna semplicemente vero o falso.

- temperature: azione che permette all'utente di ricevere informazioni sulla temperatura della casa. Richiama il metodo (getTemperature) e ritorna il valore in °C.

- sensorInfo: raggruppa tutte le informazioni precedenti in un unico messaggio.

- getWeather:invia all'utente informazioni sulle condizioni meteo attuali oppure su 3 giorni. Questa richiama, a seconda della scelta dell'utente, ("getExternalWeather") oppure ("getForecast") e ritorna all'utente le previsioni meteo.

- setLights: permette di impostare gli orari in cui accendere le luci in modo automatico. Quest'azione scrive su un file di testo informazioni sul range di tempo definito dall'utente come parametro dell'azione (millisecondi trascorsi dalla mezzanotte). Il thread di controllo delle luci legge periodicamente da questo file per controllare se il tempo attuale rientra in uno dei range definiti.

- setLightsReset: elimina le regole impostate in precedenza, ovvero cancella completamente il file contenente i range di tempo.

- autoModeOn:attiva la modalità automatica di gestione del sistema di accensione luci e regolazione luminosità.

- autoModeOff: disattiva la modalità automatica di gestione del sistema di accensione luci e regolazione luminosità.

- setComfort: azione che permette all'utente di impostare la temperatura secondo lui ideale da mantenere all'interno dell'abitazione. Il thread di controllo della temperatura si basa su questo parametro e regola l'accensione di una stufa elettrica in modo da mantenere la temperatura della stanza vicina a questo valore.

Tutte le operazioni che riguardano le lampadine Philips Hue sono gestite nel package hueController, così come l'interazione con i dispositivi della rete Zwave (sensori e prese) è gestita nel modulo zWaveController.
Questi moduli espongono delle semplici funzioni come "lightsOn" o "getSensorMeasurements" e al loro interno implementano tutta la serie di operazioni necessarie per offrire questi servizi.

2.2.2. HueController

L'interazione con il bridge Hue avviene tramite chiamate HTTP REST, quindi GET e (in questo caso) PUT, con il body della richiesta che è un testo JSON.
Per esempio, per implementare l'operazione di accensione della luce, i passi sono i seguenti (assumendo di conoscere indirizzo IP del bridge e di aver ottenuto il token che rappresenta l'username):

- inizializzazione : necessaria prima di ogni operazione sulle hue. Controlla lo stato delle hue disponibili eseguendo una HTTP GET all'indirizzo del bridge delle hue, interrogandolo sullo stato delle lampadine ("<Ip del hue bridge>/api/<username>/lights/").

- La chiamata restituisce un JSON contenente le informazioni su tutte lampadine connesse. È analizzato il testo per cercare se è presente almeno una lampadina (la lampada 1). In caso di esito negativo viene subito inviata una risposta “No lights found”.

- Se la lampadina 1 è presente, viene inviata una HTTP PUT ad un URL simile al precedente, ma specifico per la lampadina ("<Ip del hue bridge>/api/<username>/lights/1/state”) e, come body del messaggio, un JSON che specifica le operazioni sulla lampadina: “{ "on” : true, “sat”:0, “bri” : 255 }". Indica di accendere la lampadina con saturazione 0 (per “azzerare” eventuali colori prima impostati) e con luminosità massima.

Tutte le altre operazioni sulla Hue consistono sostanzialmente negli stessi passi sopra elencati, con la differenza che risiede nei parametri del JSON da inviare nella PUT: i parametri disponibili sono “on”, “bri” (intensità) “sat” (saturazione) e “hue” (valore che indica il colore).
Per ricevere maggiori informazioni sulla hue, invece, si analizza nel dettaglio il primo JSON di risposta alla GET, nel quale si possono trovare, per esempio, i valori correnti dei parametri impostati.

2.2.3. ZWave Controller

L'interazione con il gateway Zwave è leggermente diversa: avviene tramite delle Api Zway disponibili direttamente in Java, quindi non è necessario eseguire alcuno scambio HTTP (non direttamente). Invece, va istanziata la classe delle Api fornendo come parametri Ip del gateway, username e password ottenuti durante l'installazione.
A questo punto è sufficiente utilizzare i metodi di questa classe. Per esempio, per ottenere le misurazioni di un sensore, è necessario:

- Interrogare le Api Zway per farsi restituire una lista di tutti i dispositivi connessi: List<Devices> getAllDevices()

- Analizzare la lista in cerca del tipo di dispositivo che interessa (SensorMultilevel) oppure, come nel nostro caso, servirsi di un identificatore sempre uguale che discrimina univocamente un dispositivo all'interno della rete Zwave. In questo modo si può sfruttare il metodo di Zway getDeviceByNodeId, passando come parametro quell'identificatore precedentemente memorizzato, per ottenere il dispositivo specifico.

- Una volta identificato il dispositivo, se presente, vengono analizzate le varie probes che contiene. Un dispositivo, visto dalla Zway Api, non è altro che una lista di probes. Una probe è un sensore (“sonda”) che misura una sola grandezza fisica specifica (oppure è un attuatore): in un dispositivo di tipo sensorMultilevel sono quindi presenti molte probes diverse e ognuna con un singolo compito, ma di sola misurazione. Per ottenere, per esempio, la misurazione di temperatura è quindi necessario scorrere tutte le probes disponibili in quel dispositivo per trovare quella di tipo “temperature”.

- Ottenuta la probe specifica che interessa, sono disponibili dei metodi che restituiscono la misurazione attuale (getMetrics().getLevel()), assieme all'unità di misura utilizzata.

La procedura per accendere una presa è molto simile: si cerca fra l'elenco dei dispositivi disponibili in cerca di uno di tipo SwitchBinary. Questo contiene sia probe di tipo SensorMultilevel sia attuatori, di tipo SwitchBinary: per le prime, scegliendo le probes power e consumption si ottengono le misurazioni di potenza e consumo; per il secondo tipo di probe si può invocare il metodo on() e off() per accendere e spegnere la presa.

2.2.4. ControlThreads

Come citato più volte, sono stati implementati due Threads di controllo. Questi si trovano all'interno del package controlThreads e ognuno è indipendente dall'altro.
Il thread ComfortControl si occupa della gestione della temperatura interna e della luminosità delle luci. LightsControl invece decide quando accendere e spegnere le luci, rispettando i range di orari definiti dall'utente.

ComfortControl raccoglie informazioni sulla luminosità interna e sullo stato attuale delle lampadine hue e regola il loro valore di luminosità intorno ad una media ideale; inoltre misura la temperatura della stanza ed accende o spegne una presa Zwave, ipoteticamente collegata ad una stufa elettrica, per raggiungere il valore di temperatura ideale impostato dall'utente.
Il meccanismo di regolazione dei due valori si basa su un ciclo di controllo a frequenze variabili: il thread parte ad alta frequenza (mezzo secondo di intervallo), misura la luce esterna e, se questa è maggiore del valore ideale, abbassa la potenza delle luci di una piccola percentuale (10%); se è troppo bassa alza la luminosità del 10%. Ripete l'operazione ogni 500 millisecondi fino ad arrivare intorno al valore ideale (con un minimo di scarto), quindi abbassa la frequenza 3 secondi. La frequenza si rialza e il ciclo di modifica delle percentuali ricomincia soltanto quando la misurazione di luce attuale non rientrerà più nel range accettabile.
La luminosità da impostare è pari a 0 se non è rilevato alcun movimento nella stanza.
Per quanto riguarda la temperatura, il funzionamento è simile: misura la temperatura interna e, se questa è maggiore del valore impostato dall'utente, spegne la stufa, altrimenti la accende. Accende anche le luci di un colore diverso a seconda della temperatura: rosse se fa caldo, blu se fa freddo, bianche se si è raggiunta la temperatura ideale. Le luci sono accese con il valore di luminosità ideale appena calcolato, e soltanto se erano già accese, perché deve rispettare gli orari di luci forniti dall'utente o comunque deve essere già stato l'utente ad accendere la luce manualmente.

LightsControl controlla periodicamente, con una frequenza fissa di 3 secondi (precisione), se l'orario attuale rientra in uno dei range definiti dall'utente per l'accensione automatica delle luci. Legge da un file, in cui sono stati precedente scritti questi valori (una riga per ogni range), e controlla riga per riga se i millisecondi trascorsi dalla mezzanotte attuali sono tra il minimo e il massimo letti. Se trova almeno una riga che coincide, e se l'utente ha azionato la modalità automatica, allora accende le luci. Se invece non trova neanche una riga che coincide con il tempo attuale, allora spegne la luce, se era accesa e se la automode era attiva (altrimenti è stato l'utente ad accenderla quindi deve restare accesa).
E' possibile resettare tutti gli orari salvati tramite un apposito comando (resetlights), il quale provoca la cancellazione dell'intero contenuto del file.

2.2.5. WeatherApi

Un modulo dedicato all'interazione con il servizio cloud di Apixu.com per la ricezione di dati riguardanti il meteo. L'utente può richiedere le condizioni meteo attuali oppure le previsioni per i prossimi tre giorni.
Il funzionamento di questo servizio si basa su chiamate HTTP GET, con i parametri che riguardano la località e il numero di giorni della previsione direttamente specificati nell'URL. Una volta eseguita la get, viene restituito un oggetto JSON che è analizzato e parsificato per ottenere le informazioni riguardanti: condizione meteo (sole, pioggia...), temperature massime e minime e orari di alba e tramonto.
Per la condizione meteo attuale è estratta soltanto la temperatura e la condizione meteo.

3. COMPONENTI UTILIZZATI

Per poter utilizzare questo servizio è necessario avere:

- 1 Lampadina Philips Hue
- 1 Stufetta Elettrica
- 1 Multisensore Z-Wave
- 1 Presa Z-Wave
- 1 Portalampade
- 1 Raspberry Pi 3 (RPi 06)

4. SCELTE

Alcune scelte sono state prese per semplicità ed astrazione rispetto ad un sistema complesso e più efficiente, che, però, avrebbe richiesto molto più tempo per la sua implementazione ed anche molte più occasioni per testare l'interazione fra i vari componenti fisici.

Una prima scelta limitante è stata quella di trattare una sola lampadina ed una sola presa, ma soltanto per semplicità nella realizzazione di un "prototipo": l'ampliamento del progetto a più componenti, infatti, richiede soltanto la modifica di alcune funzioni nei moduli hueController e zWaveController per utilizzare tutti i componenti disponibili insieme; invece, dare un comportamento individuale ai singoli componenti uguali tra loro, è un lavoro più oneroso e tocca anche gli altri moduli.
Una scelta simile è stata quella di trattare una sola stanza della casa: anche in questo caso, per ampliare a più stanze e trattare ogni singola stanza in modo individuale, prendendo le misurazioni isolate alla stanza ed attivando i componenti, sarebbe stato necessario un tempo molto maggiore.

Non si è tenuto troppo conto di aspetti quali efficienza e consumo: per esempio si è scelto di far girare ComfortControl ogni 500 o 3000 millisecondi, anche se la temperatura non ha delle variazioni tanto evidenti in così poco tempo, questo perché il thread che controlla i due aspetti è il medesimo, e la luce potrebbe cambiare rapidamente.

Si è notato che il multi-sensore, quando interrogato, invia al controller un certo numero di informazioni, che rimangono memorizzate. Il programma quindi interagisce quasi solamente con il controller leggendo le informazioni che questo ha memorizzato; però in certi casi il buffer delle richieste non viene svuotato, oppure è molto pieno e non è possibile (per ora) svuotarlo, e l'effetto che causa è ricevere sempre la stessa informazione anche a distanza di ore. Per esempio una volta accesa una lampadina e puntata contro il sensore, ci si aspetta che il valore di luminosità letto aumenti: invece in certe occasioni, invece, rimane lo stesso per ore, e cambia soltanto dopo un certo numero di richieste.
Magari questo comportamento è causato da un eccessivo numero di richieste da parte del thread di controllo temperatura, che gira ogni 500 – 3000 millisecondi, ma purtroppo non c'è stato abbastanza tempo per indagare e quindi per ora il risultato è che l'intensità delle luci non viene regolata correttamente, perché viene rilevato sempre un valore troppo alto o troppo basso, e costante.

5. LIMITAZIONI

Purtroppo questo servizio ha alcune importanti limitazioni:

- è ridotto ad una luce eduna sola presa, perciò non gestisce effettivamente una casa
- non è possibile attivare e disattivare manualmente (tramite Telegram) la presa collegata alla stufetta
- la soluzione finale non comprende la gestione di comandi vocali
- sarebbe più pratico collegarsi ad un termostato per la gestione della temperatura, ma quelli comunemente diffusi non forniscono nessuna API
- l'uso della stufa elettrica, per come è stato studiato il sistema, porta a consumi molto elevati, in quanto la temperatura viene mantenuta intorno al valore ideale anche quando non c'è nessuno in casa
- il valore di luminosità ideale non è, per ora, settabile dall'utente.
- Il valore di temperatura ideale non è stato reso persistente quindi se il sistema fallisce e deve riavviarsi, bisogna reimpostare tale valore.
- Non è stato possibile testare il sistema per più di qualche ora consecutiva.

6. VIDEO E ALTRE RISORSE

Il video di presentazione richiesto relativo al progetto si trova all'URL: http://youtu.be/S8hGjPCVPiA.

Nella root del progetto, sotto la directory src/main/java si trovano titti i file sorgenti. Nella cartella resources sono presenti il jar compilato del programma e l'export dell'agente conversazionale di Api.Ai

Le slides di presentazione utili per l'esame orale si trovano nella cartella SLIDES.