Alma Mater Studiorum · Università di Bologna

SCUOLA DI SCIENZE
Corso di Laurea in Informatica

Applicazioni mobili ed Internet delle cose: Progettazione e sviluppo di un framework per ottimizzare l'efficienza energetica di una Smart Home

Relatore:
Chiar.mo Prof.
Marco Di Felice

Presentata da:
Gianmarco Ascenzo

Sessione II
Anno Accademico 2014-15

Indice

Introduzione

Oggi, nel contesto in cui viviamo, circondati da tecnologie che migliorano la nostra qualita' di vita, abbiamo raggiunto traguardi mai toccati nel passato. Siamo supportati da elettrodomestici delle più svariate categorie e tipologie, che ci aiutano nella vita di tutti i giorni. L'utilizzo a volte eccessivo o improprio comporta per molti utenti inesperti o poco informati, pesanti ricadute in materia di consumi, manutenzioni , costi e ancor peggio, non possiamo non considerare l'impatto ambientale peggiorato dagli spechi. La domotica tratta anche queste problematiche e studia le tecnologie che potrebbero migliorare la qualita' di vita negli ambienti domestici. Qualche anno fa, installare un impianto di domotica aveva un prezzo notevole, considerando che era necessario dover progettare l'impianto ad-hoc, per ogni singolo appartamento, in base all'esigenze dell'utente. Negli ultimi anni, per far fronte agli alti costi dell'installazione, sono state commercializzate svariate tipologie di dispositivi, che possono essere installati e gestiti in modo semplice tramite smartphone o personal computer. Queste innovazioni però possono considerarsi solo parziali perchè, molti di questi dispositivi permettono solo una gestione remota e non un aiuto concreto nell'uso efficiente delle apparecchiature a disposizione, con spesso difficolta' nell'utilizzo causato dalle differenti tipologie di sistemi che li gestiscono. Con l'evoluzione delle tecnologie all'interno delle case sono nati i primi sistemi di risparmio intelligente che permettono di razionalizzare i consumi evitando inutili sprechi, con l'utilizzo di tecnologie non invasive. Purtroppo la maggior parte di questi sistemi, data la loro semplicita' di installazione, permette solo una supervisione dei consumi senza la possibilita' di interagire con gli elettrodomestici, fornendo all'utente dei semplici consigli. Esistono pochi servizi attualmente disponibili ed i piu' affidabili hanno bisogno di installazione da parte di personale qualificato o necessitano una minima conoscenza tecnica. Riuscire a controllare, ed allo stesso tempo gestire in modo corretto ed efficiente, in base alle proprie esigenze, un sistema domotico sembra essere un obiettivo complicato da raggiungere ed in alcune occasioni anche molto costoso. Questa tesi fornisce i principi per la progettazione e l'implementazione di un

framework modulare capace di interagire con ogni tipo di scheda elettronica o dispositivo tele-leggibile. Nel capitolo 1 vengono mostrati alcuni sistemi di domotica e le differenze sostanziali con il progetto sviluppato. Nel capitolo 2 vengono visti in dettagli i requisiti minimi che un sistema di domotica deve soddisfare e quali sono le funzionalita' indispensabili per un aiuto concreto in termini di risparmio ed efficienza. Il capitolo 3 parla dell'infrastruttura del sistema, della sua architettura e dei processi principali da gestire tramite l'uso di uno schedulatore specifico. Il capitolo 4 fornisce informazioni utili per l'implementazione dell'intero sistema, dalle tecnologie utilizzate all'implementazione parziale del parser delle automazioni. Il capitolo 5 parla delle principali tecnologie utilizzate nell'implementazione del client Android, le sue funzionalita' e come avviene l'interazione con il sistema. Il capitolo 6 fornisce dati e informazioni riguardanti la correttezza delle operazioni effettuate dai task che si occupano del reperimento, della manipolazione e della storicizzazione dei dati. Il capitolo 7 tratta le possibili estensioni che potrebbero migliorare l'efficienza del sistema e dei dispositivi collegati.

Capitolo 1

Stato dell'arte

1.1 Progresso tecnologico e Iot

Negli ultimi anni stiamo assistendo alla nascita di nuove teconologie che potrebbero migliorare o addirittura rivoluzionare la qualita' della nostra vita. La diffusione di varie tipologie di apparecchi elettronici ha permesso di rendere piu' semplici e smart i dispositivi che tutti noi conosciamo, creando dispositivi general purpose in grado di connettersi tra loro tramite differenti mezzi di comunicazione. Nello specifico, un argomento di particolare interesse è l'Iot (Internet Of Things), detto anche "internet delle cose" che consiste nell'estensione di internet al mondo degli oggetti[4]. Non piu' una visione della rete che connette gli utenti tra loro ma una visione di macchine che si scambiano informazioni. Nonostante la sua enorme diffusione e ricerche inerenti alle sua flessibilita', l'Iot presenta ancora delle incoerenze sul sistema dal quale è supportato. Alla nascita di internet non si poteva infatti immaginare che un giorno la rete sarebbe cresciuta in maniera esponenziale e non si poteva certo supporre che lo spazio di indirizzamento di tutti i dispositivi attualmente utilizzato fosse 'piccolo' rispetto al numero di dispositivi che potevano far parte della rete. Per far fronte a questo problema gia' dal 2004 sono state rese disponibili versioni evolute del protollo ip in grado di indirizzare un numero estremamente piu' elevato di dispositivi, rispetto alla versione precedente[13]. La semplicita' di integrazione in reti gia' esistenti e la possibilita' di aggiungere dispositivi senza dover alterare le componenti che formano l'infrastruttura, fa dell'Iot il migliore "alleato" delle migliaia di apparecchi elettronici che vengono oggi prodotti. Ci riferiamo ad un tipo di infrastruttura che permette di collegare in rete dispositivi di ogni tipologia sebbene dotati di sensori con minor potenza di calcolo (quindi minori consumi) rispetto alle macchine che attualmente navigano su internet. L'idea

sostanziale è quella di risparmiare sul costo di produzione di ogni singolo nodo della rete, grazie al basso costo dell'hardware, e ad utilizzare in modo efficiente l'infrastruttura sulla quale si appoggia. Questo si può ottenere utilizzando una sorta di pubblicazione di informazioni, lette solo da chi è interessato, riducendo al massimo gli sprechi[14]. In certe situazioni però è molto difficile riuscire a costruire una rete di dispositivi che dialogano tutti nella stessa maniera. Molto spesso, infatti, alcuni di questi dispositivi forniscono gia' un metodo di gestione differente da quello principale, nascono così sistemi ibridi in grado di utilizzare differenti protocolli di comunicazione per svariate tipologie di apparecchi tele-leggibili[9]. In questo caso però è necessario dover apportare modifiche ad-hoc per ogni tipo di infrastruttura ed anche questo può comportare considerevoli aumenti dei costi e della manutenzione. Molti di questi sistemi offrono la possibilita' di acquistare dei moduli aggiuntivi per migliorare il funzionamento del sistema stesso[8] ma hanno dei costi aggiuntivi e l'installazione e' riservata esclusivamente a tecnici specialistici, riducendo l'interesse che questo potrebbe avere sui consumatori.

1.2 Aumentare il risparmio riducendo gli sprechi

Quanto può essere importante utilizzare al meglio i nostri elettrodomestici? Una ricerca effettuata in abitazioni domestiche per un periodo di un anno, ha evidenziato sprechi che superano il 30 % dei consumi totali, a causa di disattenzione e disinformazione, mi riferisco a luci inutilmente accese, lavatrici in funzione in orari di punta, uso di climatizzatori senza reale necessita' e uso inconsapevole di dispositivi mal funzionanti.La maggior parte dei consumatori, circa l'86 %, si è dimostrata fortemente interessata ad adottare tecnologie e sistemi di controllo che sappiano ottimizzare il funzionamento e l'efficienza degli impianti domestici[12]. Alcuni sistemi nati per far fronte a questa problematica (vedi figura 1.1) sono difficili da installare e complicati da utilizzare ed altri semplici nell'installazione e nell'utilizzo. I secondi sicuramente più diffusi, stanno prendendo piede nelle case dei consumatori. La grossa carenza è l'inattendibilita' delle informazioni. I sistemi basano le loro statistiche sulla disaggregazione[3] da un unico consumo generale ai singoli consumi degli elettrodomestici in uso durante il corso della giornata (vedi figura 1.2). Si deduce facilmente che riconoscere ogni tipo di dispositivo, tenuto conto della quantita' di modelli e tipologie presenti sul mercato, è davvero un' impresa ardua; usato solo per scopi industriali, si stanno sviluppando dei tool in grado di fornire informazioni sempre piu' precise anche

all'interno degli appartamenti[11]. Esistono applicazioni simili al lavoro descritto in questa tesi come l'applicazione freedomotic [5], un framework utile a gestire le moderne infrastrutture "smart". Permette l'integrazione con le recenti tecnolgie Z-Wave, Modbus RTU(seriale) e schede Arduino. Purtroppo pero' la configurazione e l'installazione del sistema richiede conoscenze informatiche ed elettroniche senza le quali non e' possibile gestire al meglio l'intero sistema ed inoltre una serie di bug contenuti all'interno dei protocolli mettono a repentaglio la stabilita' del sistema, che non si occupa di visionare lo stato dei moduli. Altri sistemi invece utilizzano soluzioni Wireless attive o passive per il risparmio domestico e per il controllo di anomalie[6]. Nel primo caso l'utente interagisce con l'infrastruttura tramite smartphone o dispositivi indossabili che permettono un risparmio energetico in riferimento alla localizzazione dell'utente; nel secondo caso non esiste una diretta comunicazione tra utente e sistema. In questo caso sara' il sistema in base alle sue abilita' a riscontrare cambiamenti, causati da presenza di utenti, tramite uso di sensori. La localizzazione passiva può comportare un costo notevole in termini di hardware da inserire all'interno degli appartamenti. Un argomento di particolare importanza e' la previsione dei consumi per ottimizzarne l'efficienza e ridurne i costi. Questo puo' essere fatto tramite l'uso di classificatori in tempo reale e con la previsione di utilizzi futuri di picchi di energia tramite l'uso di energy-meter che forniscono informazioni al sistema in grado di analizzare con differente granularita' i consumi attuali. Tramite l'uso di algoritmi di forecast e' possibile identificare e predire consumi anomali nell'ora successiva rispetto all'analisi, fornendo dei risultati interessanti nell'uso di sistemi di monitoraggio non invasivi[7]. In questa tesi verranno spiegate le principali funzionalita' dei sistemi domotici, e dei sistemi di risparmio, il modo in cui interagiscono , fino alla progettazione dell'intero sistema. Ogni tipologia di scheda elettronica tele-leggibile riconosciuta dal sistema verra' analizzata in base alla propria categoria. Ognuna di queste categorie permette di gestire ciascun possibile stato della macchina collegata. L'obiettivo è quello di dare la possibilita' ad ogni utente che lo voglia, di gestire e controllare da remoto i propri dispositivi e farlo in modo efficiente e responsabile. S'intende fornire informazioni sui comportamenti che aiutino a migliorare l'efficienza energetica nella vita di tutti i giorni, attraverso tecnologie appropriate e di facile utilizzo. Il sistema progettato, sara' in grado di riconoscere anomalie e sprechi e di fornire utili consigli sulla possibilita' di abbattere i costi, con conseguente riduzione dell'impatto sull'ambiente. Il sistema è centralizzato, quindi ogni sensore dovra' essere in grado o di ricevere richieste o di inviare eventuali variazioni tramite una sorta di notifica, in base al protocollo usato.

Figura 1.1: Funzionalita' di un sistema di domotica [10]

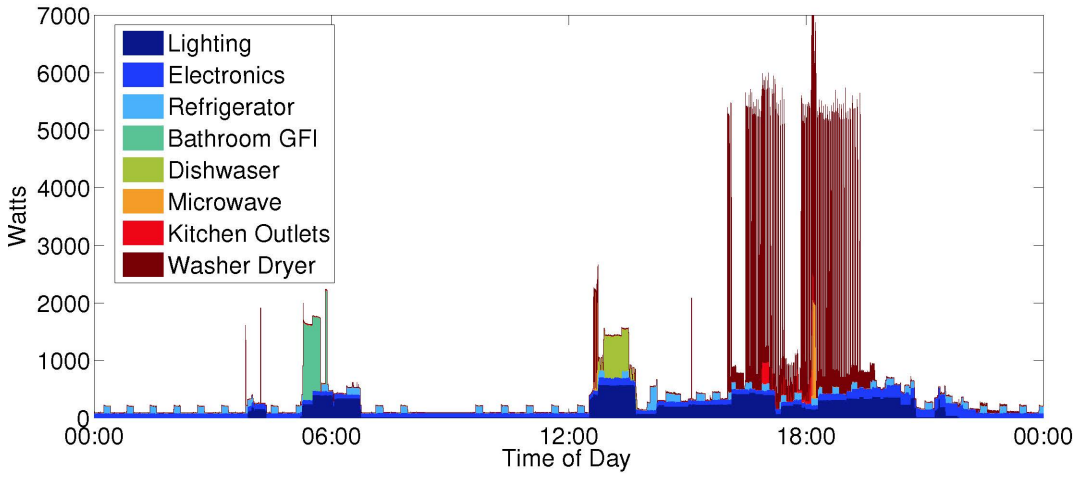

Figura 1.2: Esempio di disaggregazione [3]

Capitolo 2

La domotica e le sue funzionalita'

I requisiti che un sistema di domotica deve soddisfare, per potersi chiamare tale, potrebbero variare in base alle esigenze alle quale deve far front e agli investimenti ai quali si è disposti; dal gestire il condizionamento ambientale alla gestione di un sistema anti-intrusione. Alcuni di questi requisiti però, sono essenziali per ogni tipologia di impianto e quindi posso essere definite come funzionalità di base, tra queste troviamo:

1. **Stabilita'**: il sistema deve restare attivo anche in caso di guasti o errori evitando situazioni di pericolo e mantenendo uno stato di "equilibrio" tale da poter eseguire almeno, le funzionalita' di base evitando crash. E' possibile fornire metodi agli utenti per la risoluzione dell'errore che sarà catalogato e specificato all'interno della documentazione fornita.

2. **Sicurezza**: non deve permettere l'accesso ad intrusi o manomissione da parte di terzi, utilizzando meccanismi di autenticazione e crittografia in grado di proteggere il sistema da possibili minaccie.

3. **Compatibilita'**: deve permettere una facile integrazione con sistemi già esistenti e l'interazione con differenti tipologie di macchine, anche successivamente all'installazione.

4. **Flessibilita'**: ideato per implementare nuove funzionalita' senza comportare modifiche sostanziali, fornendo semplici metodi per l'adattamento alle differenti tipologie d'uso.

2.1 Un sistema intelligente ed efficiente

Per poter costruire un sistema efficiente sotto il punto di vista dei risparmi energetici, è necessario far uso di sensori in grado di rilevare i consumi in tempo reale e di fornirli al sistema. Diventa così necessario definire almeno due tipologie di schede elettroniche tele-leggibili che permetteranno di controllare e supervisionare l'andamento dei consumi durante la giornata. Il primo sensore verrà collegato al contatore principale e rileva il consumo totale. Il secondo può essere collegato ad ogni tipo di elettrodomestico, ne rileva il consumo e fornisce la possibilità del controllo da remoto. Le due tipologie possono essere integrate facilmente col sistema, se fanno uso del protocollo MQTT(Moquette) o REST (Representational State Transfer). Sara' inoltre possibile aggiungere nuovi protocolli di comunicazione che permetteranno di abilitare il sistema a leggere dati di sensori non riconosciuti in precedenza. Il sistema sara' formato da una parte centrale detta nucleo o core si occuperà di elaborare, storicizzare i dati e di caricare a run-time i moduli dei protocolli di comunicazione. Nello specifico il sistema sarà in grado di:

1. Fornire delle API per la gestione degli elettrodomestici collegati, con la possibilità di aggiungerli o rimuoverli senza necessita' di riavvio.

2. Fornire differenti tipologie di notifiche all'utente, ad esempio possibilita' di risparmio in base al clima e alle fasce orarie dichiarate.

3. Informare l'utente della presenza di anomalie sui consumi.

4. Mantenere traccia delle informazioni riguardanti gli utenti e della loro presenza all'interno dell'abitazione.

5. Spegnere gli elettrodomestici dimenticati accesi evitando sprechi e pericoli.

6. Spegnere gli elettrodomestici scelti in base a priorita' stabilite, nel caso in cui il consumo totale superi il limite imposto dal contratto.

7. Impostare un limite di ore/energia giornaliera per ogni dispositivo riconoscendo così difetti o guasti dell'elettrodomestico collegato.

8. Impostare delle soglie di allarme, oltre le quali è possibile disattivare o riattivare l'elettrodomestico indicato e/o ricevere delle notifiche.

9. Generare delle automazioni tramite parser IFTT (if then that) ed eseguirle in momenti stabiliti.

10. Calcolare a priori la media dei consumi mensili di ogni elettrodomestico e del totale generale.

11. Fornire all'utente la possibilità di impostare delle percentuali di risparmio su ogni elettrodomestico definendo i limiti di energia quotidiani.

12. Fornire all'utente informazioni sullo stato del sistema ogni qual volta lo si desideri.

2.2 Machine learning per il risparmio energetico

Tutti i consumi ricevuti ed elaborati dal sistema verranno controllati da un gestore di eventi, che si occuperà di classificare tramite algoritmi di machine learning le informazioni ricevute e determinare se l'azione riconosciuta è o meno un'azione corretta. In base al risultato si comporranno, se necessario, le notifiche per l'utente. Testando differenti tipologie di algoritmi, su modelli già esistenti e su altri ricavati, è possibile estrarre dei risultati attendibili in base al tipo di dispositivo che si sta analizzando. Per la gestione delle anomalie viene utilizzato un algoritmo di regressione linare, applicato a modelli che vengono continuamente generati dal sistema. Questa classificazione permette di ricavare informazioni su possibili consumi in particolari orari della giornata. In maniera diversa verrano analizzate in gruppo le diverse tipologie di eventi al fine di consigliare le possibilita' di risparmio. In questo caso infatti il sistema si avvale dell'uso di due modelli già esistenti che permettono di classificare correttamente le azioni errate che potrebbero causare sprechi. Invece per la generazione della valuazione dei consumi degli elettrodomestici, si fa uso di algoritmi di previsione tramite regressione lineare o support vector machine che permettono di generare priezioni temporali al fine di consentire la valutazione dei consumi nel tempo e quindi la previsione della spesa finale.

Capitolo 3

Progettazione di un sistema di domotica

Per progettare un sistema di domotica è necessario conoscere la composizione dell'infrastruttura e come questa coopera con i dispositivi. Suddividiamo quindi:

1. **Il nucleo o core** che si occupera' dell'elaborazione dei dati e del mantenimento della rete.

2. **Sensori general purpose**, schede elettroniche tele-leggibili che permetteranno il controllo e la gestione remota degli elettrodomestici.

3. **I client**, che forniranno all'utente informazioni inerenti allo stato del sistema e dei sensori.

Ognuna rappresenta parte essenziale del sistema, che in assenza di questi supporti, non sarebbe in grado di funzionare e di svolgere a pieno le sue attivita'(vedi figura 3.2).

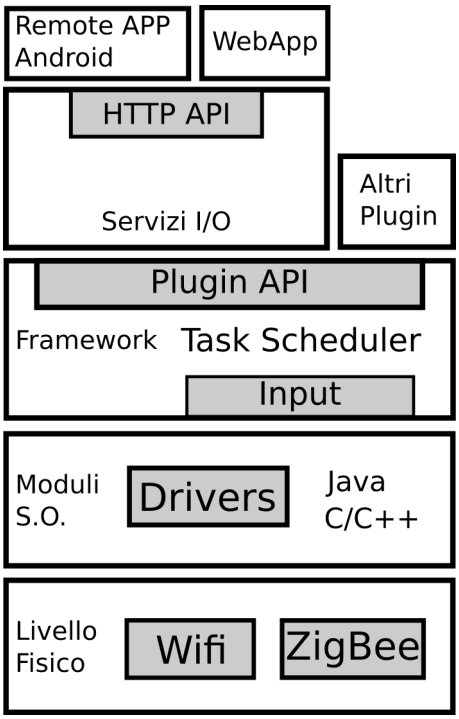

Figura 3.1: Architettura del sistema.

Processi del sisistema:	Plugin executor	Data logger	Altri processi
Gestore dei processi:	Task Scheduler		
Monitor del sistema:	WatchDog		

Tabella 3.1: Architettura del framework

3.1 Le principalid attivita'

Il core del sistema si dovra' occupare di avviare e gestire i processi principali supervisionando nel frattempo tutti i processi attivi. In questa sezione verranno mostrati i principali servizi forniti dal sistema.Ognuna di queste attivita' può essere vista come un singolo processo semi-indipendente che viene avviato in tempi prestabiliti o al verificarsi di eventi. Durante le elaborazioni vengono effettuati differenti controlli che in alcuni casi potrebbero interessare l'utente come anomalie, allarmi o possibili risparmi riguardanti dispositivi o rilevati tramite algoritmi di intelligenza artificiale. Dopo aver analizzato i dati, questi vengono storicizzati su database così da mantenere traccia di tutti i consumi passati e usarli successivamente come modelli di analisi.

3.1.1 Caricare le configurazioni

Il sistema deve essere in grado di caricare la configurazione degli elettrodomestici installati; è possibile che il file contenga anche parte della configurazione dei moduli/plugin che contengono i protocolli. I dispositivi verranno riconosciuti univocamente tramite un Tag che rappresenta il nome del dispositivo stesso. Durante la prima installazione tramite smartphone dovra' essere possibile configurare in pochi passaggi il sistema e modificarlo al bisogno, quindi ogni parametro configurato potra' essere aggiunto, rimosso e modificato.

3.1.2 Monitorare i processi

Ogni processo deve essere controllato dal sistema. E' necessario progettare un processo che controlli frequentemente la stabilita' del sistema (uso della memoria e delle risorse) e dei moduli contenenti i protocolli di comunicazione, con controllo dell'uso del disco, controllo dei processi attivi e del loro stato. Dovra' essere possibile bloccare i moduli che fanno un uso improprio della memoria o delle risorse disponibili.

3.1.3 Gestire le richieste

Per poter elaborare i dati inerenti ai dispositivi il sistema dovra' essere in grado di effettuare richieste dei valori dei dispositivi alle schede elettroniche tramite comandi di lettura e scrittura. Ogni 5 secondi dovra' inviare ai moduli delle richieste specifiche, richiedendo i tag dei dispositivi che gli interessano. Nel modo inverso, invece, in caso di scrittura, fara' una richiesta con il valore da modificare ed il tag del dispositivo. Ogni richiesta potrà avere o meno effetto sul dispositivo. Ogni valore è associato ad un intero che ne rappresenta la qualita'.

3.1.4 Elaborare i dati

Per far si' che un sistema di domotica sia in grado di interagire con i dispositivi al quale è collegato è necessario che tutti i dati che transitano vengano elaborati in base alla loro tipologia (vedi sezione 3.2) . Sara' possibile quindi:

1. fornire il consumo degli elettrodomestici non monitorati dal sistema tramite schede elettroniche. In particolare, in base all'installazione ed al

tipo di'infrastruttura, dovrà essere possibile generare o il consumo totale, tramite somme dei consumi di ogni apparecchio, oppure un consumo parziale tramite differenza del consumo totale con i singoli dispositivi.

2. fornire informazioni riguardo gli interruttori. Nel dettaglio, il sistema dovrà calcolare la durata di accensione di ogni singolo elettrodomestico, mantenendo un contatore giornaliero e mensile di questi valori. Inoltre dovra' mantenere memorizzato il momento dell'ultima/o accensione/spegnimento cosi' da poterla fornire, se richiesta, dai client.

3. fornire dettagli riguardo gli stati dei sensori. Poiche' esistono differenti tipologie di schede elettroniche è necessario che l'utente possa, sia impostare delle allarmi per esempio nel caso di sensori di movimento, sia monitorarne lo stato per esempio la temperatura ambientale. In questo caso sara' necessario inserire manualmente le soglie minime e massime tollerate oppure si potra' selezionare il tipo di elettrodomestico tramite differenti tipologie pre-configurate sul sistema.

3.1.5 Generare le notifiche

E' necessario progettare un meccanismo che permetta di mandare notifiche riguardo il sistema. Dall'informare l'utente su come poter risparmiare, fino ad informarlo riguardo anomalie riscontrate dal sistema. Ogni tipo di notifica è mappata da una chiave univoca e può sia consentire differenti scelte all'utente riguardo l'evento notificato sia notificare un semplice consiglio ecologico in base al clima.

3.1.6 Gestire il database

Il sistema deve permettere la scrittura di dati su database tramite mutua esclusione evitando accesso concorrente e consentire invece la lettura simultanea dei dati. Dovrà permettere in oltre in modo semplice l'interazione di nuovi processi col database fornendo metodi adatti all'apertura, chiusura, lettura e scrittura senza compromettere' l'integrita' dei dati.

3.1.7 Caricare e gestire le automazioni

E' necessario che il sistema si occupi di gestire le automazioni con la possibilita' di creare, gestire, avviare ed interrompere azioni pianificate temporalmente su due code distinte, quella del sistema e quella degli utenti. La coda del sistema è la coda dei processi che il sistema utilizza ad ogni avvio di un nuovo processo. La coda degli utenti invece è dinamica e varia

in base al numero di automazioni impostate dall'utente o create dal sistema per la gestione efficiente degli elettrodomestici, come per esempio delegare al sistema l'accensione della lavatrice. Questo creerà un'automazione che accenderà la lavatrice all'iniziare della migliore fascia oraria e questa automazione verra' inserita nella coda degli utenti. Sara' possibile tramite smartphone aggiungere o rimuovere automazioni senza dover riavviare il sistema.

3.1.8 Gestione consumi extra e sprechi

E' importante che esista un processo che controlli che il consumo totale sia inferiore al limite imposto dal contratto.Questo processo permetterà di evitare i distacchi della fornitura causati dal superamento dei consumi. Sara' possibile configurare una lista ordinata di dispositivi da spegnere automaticamente e progressivamente nel caso in cui i consumi superino le soglie contrattuali di disponibilità , oppure sara' possibile far decidere all'utente quali dipositivi spegnere, tramite un menu disponibile all'arrivo della notifica. Se il processo entrerà in uno stato di allarme poichè non è riuscito a spegnere un numero sufficiente di elettrodomestici da ridurre il carico e' possibile che la corrente venga a mancare. In questo caso il sistema dovra' interrompere qualsiasi scrittura sul disco per evitare i danni dovuti all'interruzione dell'energia elettrica. Il processo tornerà a funzionare alla rialimentazione dell'impianto. Inoltre dovra' controllare e gestire gli eventi riguardanti gli utenti. Con l'uso dello smartphone sarà possibile spegnere gli elettrodomestici associati all' uscita dell'utente, nel caso in cui lo stesso non si trovi in casa. Se nessun elettrodomestico è associato all'utente ma sarà rilevato uno spreco, sarà possibile che il sistema mandi una notifica informando sulla possibile dimenticanza e la richiesta di spegnimento da remoto.

3.1.9 Possibili sotto servizi

Puo' essere efficace aggiungere un servizio di previsioni del meteo che si occupi di mantenere aggiornate le informazioni per poter fornire al sistema notizie precise su eventuali precipitazioni, o caldo eccessivo, o abbassamento della temperatura, e così via. Tramite algoritmi di intelligenza artificiale sara' possibile ricavare utili consigli di risparmio da notificare all'utente. Un ulteriore servizio può controllare i cambiamenti dell IP (in caso di IP dinamici) e notificare in modo del tutto trasparente la modifica al client, cosicche' il client sia in grado di raggiungere il sistema da remoto anche senza l'uso di DNS.

3.2 I moduli e l'integrazione dei protocolli

Una parte essenziale del sistema saranno i plugin o moduli, che implementeranno i protocolli di comunicazione con le schede elettroniche tele-leggibili, astraendo l'elaborazione dei dati e la loro manipolazione dal modo in cui questi vengono reperiti. Non esiste un numero limite di moduli che il sistema potra' caricare. L'installazione dovra' essere molto semplice ed in futuro, si potrebbe prevedere l'uso di un market online. Per tutti gli sviluppatori che ne hanno il bisogno sara' possibile utilizzare l'SDK per aggiungere nuovi protocolli al sistema senza alcuna ricompilazione. Sarà quindi necessario implementare l'SDK fornendo allo sviluppatore metodi per interagire con il sistema, tramite interfaccie comuni. Ogni plugin dovrà generare un manifest contenente il nome del pacchetto della classe principale e dopo la compilazione dovrà essere copiato all'interno della cartella plugins. Senza questi, il sistema non riuscirebbe a comunicare con i dispositivi. E' fondamentale che i plugin siano stati testati e abbiano un'ottima stabilità altrimenti il sistema, in caso di uso spropositato della memoria, dovrebbe escluderli dalla coda delle richieste. Le funzionalita' principali dei moduli saranno quelle di:

1. Effettuare richieste di lettura e scrittura alle schede elettroniche
2. Effettuare richieste di lettura e scrittura al sistema
3. Leggere la propria configurazione dal file di configurazione principale, ove necessario.

Ogni plugin comunicherà con svariate tipologie di sensori differenti, per far sì che il sistema sia in grado di riconoscere e gestire qualunque apparecchio esistente, ognuna di queste viene divisa in tre classi di schede elettroniche tele-leggibili:

1. **Misuratore di consumi** Una scheda elettronica in grado di rilevare la potenza istantanea consumata dall'apparecchio.
2. **Interruttore analogico/digitale** Una scheda elettronica dotata di interruttore analogico e digitale OFF/ON;
3. **Sensore analogico/digitale** Una scheda elettronica dotata di un sensore di input analogico o digitale 0/1;

Ogni tipologia seguira' il proprio flusso di elaborazione, alcuni dispositivi verranno analizzati in base ai loro consumi altri in base ai loro stati, evitando così controlli inutili. Attualmente vengono utilizzate delle schede elettroniche "fatte in casa" (vedi figura 3.4).

Energy-meter, sensori, Inverter, ecc.
Mqtt, Rest, Modbus, ecc.
Plugin
Core

Tabella 3.2: Architettura dell'infrastruttura

3.3 Gestione tramite Android

L'ultima componente del sistema e' un'applicazione destinata alla piattaforma Android. Questa applicazione sarà in grado di gestire e controllare completamente il sistema tramite smartphone, mostrando informazioni sui consumi attuali, sullo stato degli utenti, sulle buone regole da seguire. L'applicazione sarà inoltre in grado di dare informazioni dettagliate su anomalie o possibili guasti, mettendo a disposizione del client differenti funzionalità che possano delegare al sistema il controllo degli apparecchi. L'applicazione dovra' fornire un interfaccia user-friendly per la gestione dell'intero sistema. In questa sezione verranno trattate le principali funzionalita' da mostrare all'utente. Ognuna di queste tipologie avrà la propria interfaccia grafica e di interazione con l'utente. Esse verranno storicizzate sullo smartphone e rese disponibili in una sezione apposita.

3.3.1 Visualizzare lo stato degli elettrodomestici

Tali informazioni riguarderanno l'andamento dei consumi e saranno visualizzabili con diverse icone in base all'uso che si sta facendo del dispositivo.Per ogni dispositivo sarà possibile conoscere il momento dell'accensione e dello spegnimento, ed ogni dettaglio inerente al suo tempo di utilizzo, alla quantità di energia consumata, alla durata media di accensione. Si riceveranno, inoltre, consigli utili per un migliore utilizzo tramite notifica .

3.3.2 Mostrare informazioni sugli utenti

Previa consenso, sarà possibile per l'utente, spedire in modo del tutto automatico informazioni al sistema riguardanti la sua presenza in casa e generare una lista per i quali sarà previsto lo spegnimento automatico, quando nessuno più sarà in casa.

3.3.3 Fornire lo stato del sistema

Sarà possibile visionare lo stato del sistema per controllare eventuali condizioni di errore e processi mal funzionanti. Dovra' essere resa disponibile la funzionalita' di reboot e risoluzione degli errori da remoto.

3.3.4 Gestire le automazioni di IFTT

Dovra' essere disponibile una scheda che permetta la creazione di automazioni particolari (IFTT) associate a utenti o dispositivi; al verificarsi della condizione sara' possibile eseguire delle azioni sugli elettrodomestici. Tutte le condizioni vengono ospitate dal sistema quindi il client Android non dovra' effettuare nessun operazione se non quella di aggiunta, rimozione e modifica delle automazioni. Sara' possibile modificare lo stato degli elettrodomestici nel caso in cui per esempio un utente sia uscito o rientrato a casa, oppure ricevere notifiche in base ad eventi particolari come il cambio di stato dei sensori di movimento. Le automazioni saranno rese disponibili a tutti i client connessi allo stesso sistema.

3.3.5 Visualizzazione delle notifiche

Una schermata di log conterra' tutti i consigli notificati all'utente ma anche quelli spediti a tutti gli abitanti della casa. Dovra' essere possibile visionare ogni notifica con dettagli sull'informazione contenuta. Esisteranno differenti tipologie di notifiche accettate dai client e memorizzate nella schermata di log .

Notifiche sui risparmi

Le notifiche sui risparmi dovranno fornire delle informazioni specifiche sul possibile risparmio, per esempio: se accendi la lavatrice dopo le 21.00 potresti risparmiare 0.10 Cent./Euro l'ora. Dovranno essere disponibili 4 scelte diverse, per ognuna di queste sara' necessario notificare la decisione al sistema. La prima scelta possibile sara' quella di far gestire il risparmio dal sistema stesso, che creera' una condition e impostera' l'accensione/spegnimento programmata ad un orario in cui è possibile risparmiare. La seconda scelta potra' permettere all'utente di impostare manualmente un altro orario di accensione/spegnimento, in questo caso verra' creata una condition ma con gli orari scelti dall'utente. La terza scelta permettera' di ignorare il messaggio limitatamente a quella situazione, non effettuando alcuna operazione sull'elettrodomestico. La quarta ed ultima scelta permettera' la disattivazione della notifica in modo permanente o fino alla modifica successiva.

Notifiche sulle anomalie e consumi

Le notifiche sulle anomalie e sui consumi saranno identiche a quelle sui risparmi, l'unica differenza sara' il messaggio visualizzato che conterra' informazioni dettagliate riguardo l'anomalia o i consumi e le possibili azioni da intraprendere.

Notifiche su allarmi

Le notifiche delle allarmi dovranno essere configurate manualmente dall'utente che decide di tenere sotto controllo un elettrodomestico. Queste potranno contenere allarmi inerenti ai consumi attuali, o ai tempo di accensione dell'elettrodomestico durante il corso della giornata.

Notifiche generiche su eventi

Tramite il costruttore di automazioni dovra' essere possibile impostare delle notifiche con testo personalizzato inerenti ad eventi specificati dall'utente.

3.4 Architettura e compatibilita'

Il sistema nasce con l'intento di poter lavorare su tutte le architetture che supportano Java SE (versione >= 1.6). L'dea è quella di poter soddisfare ogni tipo di esigenza, dall'installazione su server cloud all'installazione su computer di casa. Attualmente il programma è in test sulla prima versione del raspberryPi (vedi figura 3.3), non ha bisogno di nessuna dipendenza e contiene al suo interno un web-service e un broker mqtt, attivi fin dal primo avvio. Il sistema non necessita di installazione e permette l'avvio automatico all'accensione del computer. Ogni modulo può essere implementato facendo uso dell'SDK chee fornisce i metodi principali per interfacciarsi col sistema. E' possibile aggiungere nuovi plugin semplicemente trascinandoli all'interno della cartella plugins. Le schede elettroniche attualmente utilizzate sono state create "in casa" tramite l'uso di arduino utilizzando un modulo wifi (Esp8266), un sensore di corrente, e un rele' di controllo a 220V (vedi figura 3.5). I protocolli utilizzati sono mqtt e rest ed entrambe le tipologie possono interfacciarsi col sistema, senza alcuna modifica. Sia modulo wifi che arduino contengono dei firmware ad-hoc che permettono loro di comunicare, sfruttando al meglio le capacita' dell'uno e dell'altro. Ogni sensore ha un ingresso/uscita AC. All'ingresso verrà collegato l'elettrodomestico e l'uscita verrà collegata alla presa di corrente, semplice ed intuitivo. L'applicazione

mobile può essere installata su tutte le vesioni di Android maggiori o uguali alla 4.4.2 ed è necessario che il wifi sia attivo durante il suo uso in casa. L'applicazione e' in grado di collegarsi al sistema mandando dei messaggi in broadcast per conoscere il suo ip locale e remoto. Non effettua alcuna azione in background, quando è chiusa, ma è in grado di fare eseguire al sistema le azioni previste e di ricevere notifiche di tipo Push. A questo proposito è necessaria anche l'installazione di Google Play Services. L'architettura dell'intero sistema è stata pensata per essere compatibile con la maggior parte di architetture attualmente esistenti e disponibili sul mercato semplificando notevolmente l'installazione e la manutenzione.

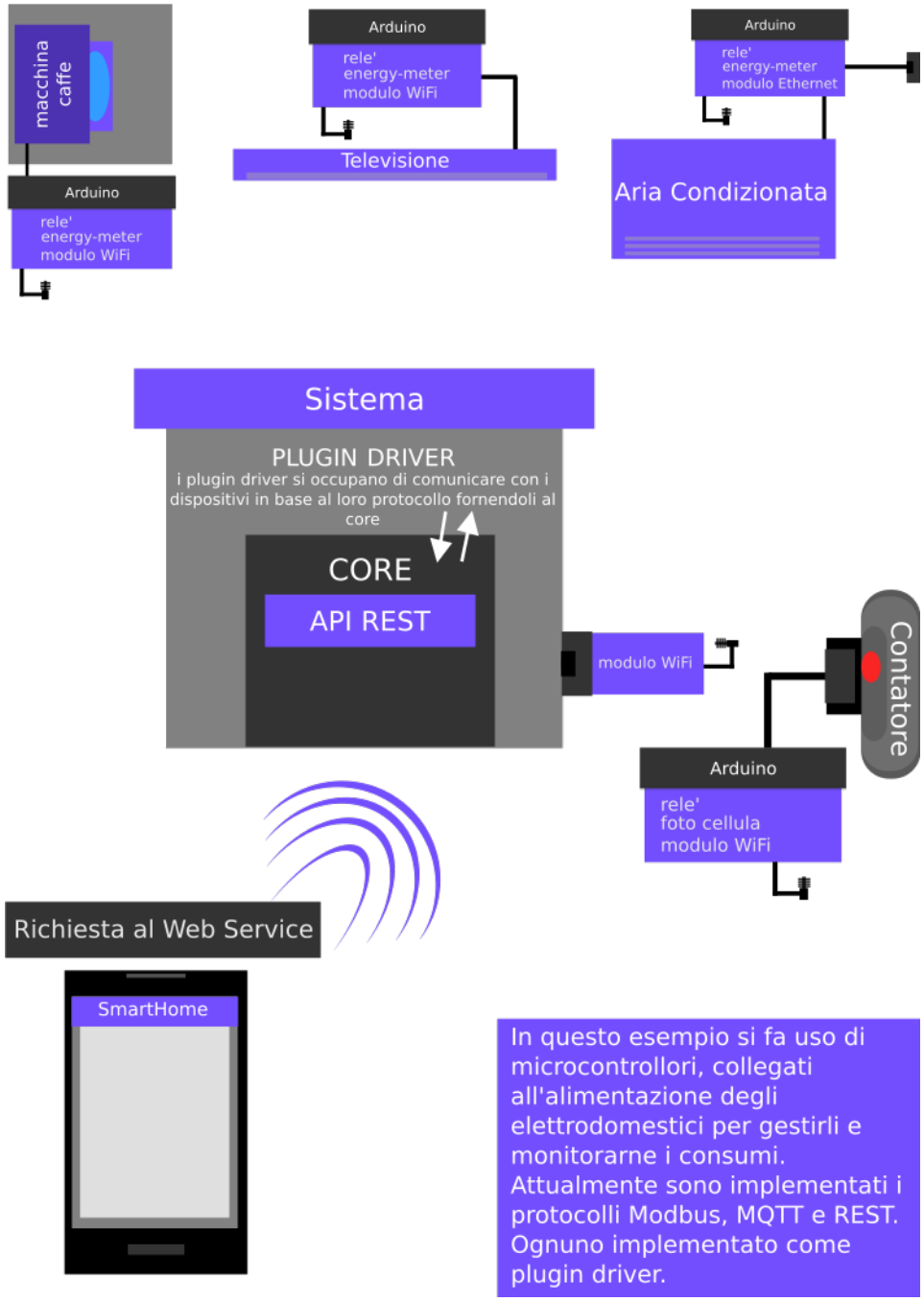

Figura 3.2: Esempio di un'installazione del framework in una rete di dispositivi MQTT e REST.

Figura 3.3: RaspberryPi utilizzato attualmente per ospitare il sistema.

Figura 3.4: Sensore attualmente in uso

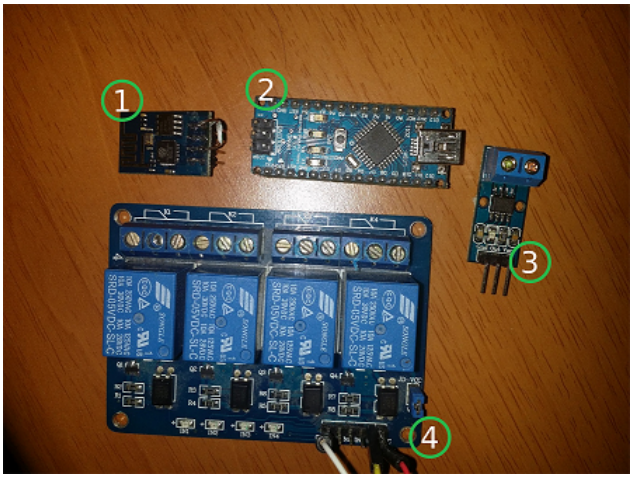

Figura 3.5: Componenti del sensore. 1: Modulo wifi Esp8266 - 2: Arduino Nano - 3: Sensore di corrente - 4: Blocco da 4 rele'

Capitolo 4

Implementazione e tecnologie

Ogni sezione di questo capitolo mostrera' le principali implementazioni dei servizi forniti dal sistema. L'intero progetto è sviluppato con Java, nato per essere il più possibile indipendente dalla piattaforma di esecuzione, aumentando la portabilita'e la flessibilita' complessiva dell'intero sistema. La gestione del progetto è delegata al software Maven che utilizza un costrutto conosciuto come Project Object Model (POM) tramite un file XML che descrive le dipendenze fra il progetto e le varie versioni di librerie necessarie.

JCron gestore dei processi del sistema
Java Reflection per il caricamento a runtime dei plugin
Tomcat Web-Server
Jersey RESTful Web-Services
Weka algoritmi di machine-learning
HyperSQL DataBase totalmente scritto in Java
Google Cloud Messaging, per le notifiche Push
Gson convertitore di oggetti Java in Json
Parser personalizzato per creare automazioni
Protocolli Modbus, MQTT, REST
JAXB convertitore da XML a oggetti Java
Joda-Time un ottimo sostituto delle classi Java date e time
SDK interamente in Java per l'implementazione di nuovi protocolli

Tabella 4.1: Tecnologie utilizzate

4.1 Discovery tramite broadcast

Durante la prima installazione e in alcuni casi successivamente è necessario fornire ai client tecniche che permettano di collegarsi al sistema senza conoscere il suo indirizzo di rete, così da favorire un installazione semplice ed immediata. Per far ciò abbiamo bisogno di un processo Server che stia in attesa di pacchetti UDP spediti su un particolare indirizzo logico, detto anche indirizzo di broadcast, permettendo così di spedire ai client il suo ip. Nel pacchetto sara' contenuto l'indirizzo ip locale e remoto del computer.

4.2 Un gestore di processi con JCron

Nello sviluppo di un sistema di domotica potrebbe essere utile integrare uno schedulatore che permetta l'avvio e l'interruzione di processi in momenti predefiniti. Per esempio potrebbe essere utile controllare ogni minuto l'andamento generale dei consumi ed invece ogni ora le previsioni del meteo. A tal proposito si può far uso di una libreria esterna, JCron, che permette l'avvio dei processi specifici tramite l'uso di pattern che indicano il momento in cui i processi verranno avviati. Lo scheduler è completo e funzionante ma è necessario apportare alcune modifiche per adattarlo al nostro scopo, come l'uso di code per tipologia di processo, evitando di unire i processi utenti con quelli del sistema, ed anche aggiungere la possibilita' di caricare una nuova tipologia di classe che potrebbe rappresentare un processo standard del sistema. D'ora in poi ogni processo sara' gestito da un unico gestore di processi ed ereditera' i metodi del processo standard.

4.3 Comunicazione con i moduli

Come detto in precedenza tutti i moduli con i quali il sistema interagisce sono contenuti all'interno della cartella plugins. Il caricamento avviene a run-time tramite reflection, una tecnica che permette di caricare una classe sconosciuta a tempo di compilazione e utilizzarla (quasi) come una qualsiasi altra classe. Ogni classe principale di ogni modulo viene caricata ed inserita in una lista che periodicamente viene scandita per caricare i moduli ed eseguire le richieste. Ogni plugin deve estendere la classe astratta DriverAdapter fornita con l'SDK che permette di creare la classe principale che interagisce col sistema. Ogni plugin eredita l'implementazione dell'interfaccia Callable che permette di eseguire un task entro un timeout. Così facendo ogni modulo, quando gli verra' fatta una richiesta di lettura o scrittura su una scheda

elettronica, non potra' far aspettare indefinitamente il sistema ma dovra' restituire qualcosa o scatenare un eccezione entro un tempo predefinito.

4.3.1 Timeout tramite intervalli di confidenza

Limitare il piu' possibile il tempo in cui un modulo possa resituire un risultato può essere efficiente dal punto di vista delle prestazioni del sistema ma può essere rovinoso nel caso in cui alcuni sensori abbiano delle alte latenze, potrebbe verificarsi una situazione in cui i tempi di risposta del sensore siano piu' alti dei tempi di risposta del modulo, escludendo così ogni possibile speranza di connessione col dispositivo. Il metodo migliore è quello di creare un nuovo processo che si occupera' di mantere un buffer FIFO con i tempi di risposta dei moduli, per l'esattezza conterra' gli ultimi 10 timeout. Ad ogni richiesta viene aggiunto in lista il nuovo tempo di risposta e tramite intervalli di confidenza viene scelto il tempo migliore entro il quale il modulo dovrebbe rispondere. Ogni qual volta che il modulo non fara' in tempo a fornire un risultato e generera' un eccezione, il massimo timeout consentito verra' aggiunto in lista permettendo cosi' sia la crescita che la riduzione del timeout.

4.4 Implementazione delle API con Tomcat e Jersey

Per poter gestire l'intero sistema è necessario fornire dei metodi ai Client che gli permettano di scambiare informazioni di diverso tipo. E' possibile creare un protocollo ad-hoc oppure sceglierne uno che possa essere adatto alla gestione di un sistema domotico. Per poter implementare al meglio tutte le possibili richieste usiamo l'architettura Rest su protocollo HTTP per lo scambio di informazioni con i client. Si fa uso di un web-service, Tomcat 8, che nella versione embedded non ha bisogno di alcuna dipendenza. Fornire però delle API (insieme di procedure disponibili) ai client tramite un web-service obbliga l'implementazione dell'intera architettura RestFul. Per facilitare l'implementazione possiamo far uso di un framework, chiamato Jersey, che permette di implementare rapidamente tutte le possibili procedure da rendere disponibili ai Client.

```
// HTTPMethods.java - esempio di uno dei metodi implementati
   tramite Jesey. Questa permette la registrazione degli utenti
  @GET
  @Path("/register")
```

```
@Produces(MediaType.TEXT_HTML) //produce testo html
public Response register( @Context UriInfo uriInfo ) {
   List<String> name = uriInfo.getQueryParameters().get("name");
   List<String> sex = uriInfo.getQueryParameters().get("sex");
   List<String> idKey =
       uriInfo.getQueryParameters().get("idKey");
   List<String> notification =
       uriInfo.getQueryParameters().get("notification");
   if(name != null && sex != null && idKey != null &&
       notification != null && name.size() > 0 && sex.size() > 0
       && idKey.size() > 0 && notification.size() > 0){
      String keyToRemove = null;
      boolean needRemove = false;
      for(User utente : SmarthomeCore.userKey.values()){
         //controlla se username o IdKey esistono e nel caso
         aggiorna l'oggetto, gestita solo l'aggiunta di un
         utente per volta i successivi non verranno presi in
         considerazione
        if(utente.getIdKey().contentEquals(idKey.get(0))){
          keyToRemove = utente.getIdKey();
          needRemove = true;
          break;
        }
      }
      if(needRemove && keyToRemove != null )
        SmarthomeCore.userKey.remove(keyToRemove);
      SchedulerManager.executeOnce(new SchedulerUsers(new
          User(name.get(0), sex.get(0),
          Boolean.parseBoolean(notification.get(0)),
          idKey.get(0)), UserEnum.SAVE_USER));
      return Response.ok().build();
   }
   else
      return Response.error().build();
}
```

4.4.1 Caricamento della WebApp

Con l'uso di Tomcat abbiamo a disposizione la possibilita' di caricare una WebApp da utilizzare sia per configurare il sistema, sia per gestirlo. Al momento non è stata implementata alcuna applicazione web ma è possibile

farla caricare al sistema aggiungendo il suo contenuto all'interno della cartella WebContent del progetto.

4.5 Gestore di eventi

Uno dei processi che permette di riconoscere particolari situazioni è il gestore degli eventi. Il gestore di eventi si occupa di analizzare tutti i dati che transitano nel sistema inerenti ai dispositivi e si occupa di:

1. Generare gli eventi, catturati poi dai processi che classificheranno possibili anomalie o risparmi.

2. Gestire e processare gli eventi.

Nel primo caso, utilizzando un interfaccia notifica l'evento generato, saranno poi gli algoritmi di machine learning che riconosceranno particolari situazioni. Nel secondo caso invece, il sistema si occupa di gestire gli eventi mediante semplici controlli su allarmi, stati o fascie orarie. Elabora le tre principali tipologie di sensori e calcola i consumi, convertendo per esempio i watt in kWh oppure azzerando le statistiche giornaliere ogni notte.

4.6 Analisi dei dati

Tutti i dati che provengono dai sensori vengono analizzati dal sistema e gestiti tramite il gestore di eventi. Nel caso in cui il gestore generi un evento, è possibile che differenti processi siano adibiti a classificare quegli eventi. Vengono analizzate tre tipologie di eventi tramite machine learning. Ognuno di questi rappresentato da una notifica di versa con messaggi e funzionalita' diverse.

4.6.1 Uso di weka

Per poter classificare correttamente ogni singolo evento è necessario testare e analizzare i risultati di differenti tipi di algoritmi di machine learning. Il software scelto per l'analisi e l'implementazione degli algoritmi è Weka, scritto interamente in java, che fornisce una semplicissima interfaccia grafica per effettuare i propri test. Weka è una collezione di algoritmi di machine learning che permettono di estrarre informazioni interessanti tramite l'analisi di dati.

4.6.2 Modelli di analisi

Per poter effettuare delle corrette previsioni di variabili incognite o di classificazioni è necessario che il nostro training set o modello sia strutturato coerentemente da permettere la generazione di un albero di possibili soluzioni che sia in grado di catalogare i nostri test set(vedi figura 4.2). In questa situazione il sistema si avvale di due tipologie di modelli:

1. Sugli sprechi. Contiene la classificazione di differenti tipologie di elettrodomestici. Ad ognuno vengono associate delle buone regole per evitare degli sprechi in base alle fascie orarie dichiarate. Ogni qual volta viene generato un evento, verra' classificato tramite questo modello. Se il comportamento associato non è una buona regola, viene informato l'utente che è possibile risparmiare, per esempio accendendo l'elettrodomestico dopo un certo orario.

2. Sui consigli. Generato anche questo a priori. Ad ogni elettrodomestico vengono associati tutte le possibili condizioni climatiche durante le quali si potrebbero fornire dei consigli agli utenti. Per esempio nel caso in cui la lavatrice passi da uno stato ON ad uno stato OFF durante una giornata calda e soleggiata viene consigliato all'utente di stendere i vestiti all'esterno e di non utilizzare l'asciugatrice.

4.6.3 Classificazione di eventi tramite RandomTree

Tutti i modelli presi in analisi, sia quelli generati a priori che quelli generati dal sistema vengono classificati tramite l'algoritmo di RandomTree(vedi figura 4.1). Infatti è possibile utilizzare successivamente,tutti gli algoritmi valutati tramite l'interfaccia grafica di Weka, ed integrarli all'interno del proprio progetto Java.

```
// metodi per la classificazione di una parte di eventi

public void switchChanged(Param param, int oldValue, int
    newValue) {
   if(getHashConfig() != null){
      if(getHashConfig().get(param.getTag()) != null){
          //accesso rapido alla configurazione del parametro
         ParamConfig conf = getHashConfig().get(param.getTag());
            if(conf.isEnableSavingAdvice()){ //se e' abilitato il
                risparmio per fasce orarie
               try {
                  checkAdvice(param, conf, newValue);
```

```
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        }
    }
}

public void checkAdvice(Param param, ParamConfig conf, int
    newValue) throws Exception {
    //carico il modello esistente
    RandomTree treeClassifier = (RandomTree)
        weka.core.SerializationHelper.read(modelEnergySave);
    FastVector fvWekaAttributes =
        getInstancesAttributesForEnergySave();
    Instances originalTrain = new Instances("Relation",
        fvWekaAttributes, 10);
    originalTrain.setClassIndex(3);
    Instance originalTrains = new Instance(4);
    // setto i dati attuali
    originalTrains.setValue((Attribute)fvWekaAttributes.elementAt(0),
        getCurrentUos().toLowerCase());
       originalTrains.setValue((Attribute)fvWekaAttributes.elementAt(1),
          STATUS);
    originalTrains.setValue((Attribute)fvWekaAttributes.elementAt(2),
        conf.getAppliance());
    originalTrains.setValue((Attribute)fvWekaAttributes.elementAt(3),
        "?"); // campo incognito
    originalTrain.add(originalTrains);
    int s1 = 0;
    //classificazione dell'incognita
    double value =
        treeClassifier.classifyInstance(originalTrain.instance(s1));
    String prediction =
        originalTrain.classAttribute().value((int)value);
    if(prediction.toLowerCase().contentEquals("bad_practise")){
       //genero e spedisco la notifica corretta per il tipo di
          predizione
       SchedulerManager.executeOnce(new
          ScheduleNotification(getDatabaseLogger(),
          notification_improves_consumption,
          generateNotificationConsumption(param.getTag(),
          prediction.toLowerCase()));
```

```
    }
  }
```

```
=== Evaluation on training set ===
=== Summary ===

Correctly Classified Instances          66               100      %
Incorrectly Classified Instances         0                 0      %
Kappa statistic                          1
Mean absolute error                      0
Root mean squared error                  0
Relative absolute error                  0      %
Root relative squared error              0      %
Total Number of Instances               66

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                 1         0          1         1         1          1        good_practise
                 1         0          1         1         1          1        bad_practise
Weighted Avg.    1         0          1         1         1          1

=== Confusion Matrix ===

  a  b   <-- classified as
 54  0 |  a = good_practise
  0 12 |  b = bad_practise
```

Figura 4.1: Esempio di classificazione con Weka tramite l'algoritmo RandomTree.

```
RandomTree
==========

class = washingmachine
|   event = ON
|   |   state = bad : bad_practise (1/0)
|   |   state = good : bad_practise (1/0)
|   |   state = excellent : good_practise (1/0)
|   event = OFF : good_practise (3/0)
class = aircondition
|   state = bad
|   |   event = ON : bad_practise (1/0)
|   |   event = OFF : good_practise (1/0)
|   state = good : good_practise (2/0)
|   state = excellent : good_practise (2/0)
class = lamp
|   event = ON
|   |   state = bad : bad_practise (1/0)
|   |   state = good : bad_practise (1/0)
|   |   state = excellent : good_practise (1/0)
|   event = OFF : good_practise (3/0)
class = fan : good_practise (6/0)
class = dryer
|   event = ON
|   |   state = bad : bad_practise (1/0)
|   |   state = good : bad_practise (1/0)
|   |   state = excellent : good_practise (1/0)
|   event = OFF : good_practise (3/0)
class = dishwasher
|   event = ON
|   |   state = bad : bad_practise (1/0)
|   |   state = good : bad_practise (1/0)
|   |   state = excellent : good_practise (1/0)
|   event = OFF : good_practise (3/0)
class = tv : good_practise (6/0)
class = boiler
|   event = ON
|   |   state = bad : bad_practise (1/0)
|   |   state = good : bad_practise (1/0)
|   |   state = excellent : good_practise (1/0)
|   event = OFF : good_practise (3/0)
class = pc : good_practise (6/0)
class = mainac : good_practise (6/0)
class = oven
|   state = bad
|   |   event = ON : bad_practise (1/0)
|   |   event = OFF : good_practise (1/0)
|   state = good : good_practise (2/0)
|   state = excellent : good_practise (2/0)
```

Figura 4.2: Esempio dell'albero generato con Weka tramite l'algoritmo RandomTree.

4.7 Implementazione di HyperSQL DataBase

Ogni valore dopo essere analizzato e manipolato deve essere salvato su database. E' possibile utilizzare database sql o no-sql. Molti database sql e no-sql non sono implementati interamente in Java ma fanno uso del framework JNI (Java Native Interface) che consente al codice Java di richiamare codice specifico di un determinato sistema operativo. Cosi' facendo non solo si hanno delle grosse perdite in termini di prestazioni ma si perde l'independenza dalla piattaforma che esegue i metodi nativi. Lavorando con con questa piattaforma quindi è necessario che il database sia interamente scritto in Java per ottimizzare le operazioni e ricorriamo allora all'uso di HSQLDB(HyperSQL DataBase). Un database sql che fornisce un numero svariato di tipi di dato e con supporto al multithread. E' necessario implementare un processo che si occupi di mantenere la mutua esclusione quando due o piu' processi tenteranno di scrivere sul database concorrentemente ma dovra' permettere la lettura simultanea. Questo processo dovra' essere avviato insieme al sistema e dovra' occuparsi di riempire il database e di mantenere aggiornati i valori dei dispositivi.

4.8 Un parser personalizzato

Poter creare delle automazioni all'interno della propria casa è uno degli interessi maggiori per ogni smanettone. Infatti è possibile creare delle automazioni causa, effetto. Per esempio se l'utente gianmarco ascenzo è a casa e sono le 9 allora prepara il caffe'. Ogni automazione è dichiarata tramite una condition, ogni condition può contenere un numero illimitato di if(Causa) e di that (effetto). L'intero sistema mantiene la lista di tutte le condition chee possono essere aggiunte e rimosse manualmente. L'avvio della verifica per ogni condition può essere specificata tramite un pattern Cron e può contenere controlli su utenti e dispositivi. Il parser è implementato con l'uso del pattern Visitor ed ogni condizione if esegue una visita, si prosegue finchè tutte le condizioni if abbiano effettuato la visita. La prima condizione if che non viene verificata interrompe l'esecuzione (lazy parsing).

Formula 4.8.1 *La verifica di ogni condizione puo' essere rappresentata come la formula:*

$$\phi = \prod_{i=1}^{n}(a_i)$$

con a = clausola della i-esima condizione e phi risultato della verifica

Attualmente possono solo essere effettuate delle assegnazioni e possono essere spedite delle notifiche. Ogni condizione avra' un operatore che specifichera'

il controllo da fare sui dati, ognuna delle due parti della condizione conterra' tipo, valore e contenuto.

```
// Resolver.java - questa classe si occupa di risolvere e
    verificare le condizioni
public class Resolver {
   Comparator comparator;
   PairResolverDate resDate;
   PairResolverParam resParam;
   PairResolverValue resValue;

   public Resolver() {
      comparator = new Comparator();
      resDate = new PairResolverDate();
      resParam = new PairResolverParam();
      resValue = new PairResolverValue();
   }

   public boolean assign(That that){ //funzione che si occupa
       dell'assegnamento del valore al parametro
         Param param = SmarthomeCore.bufferKey.get(that.getName());
         if(param == null){
            System.out.println("IFTT param null");
            return false;
         ArrayList<Param> list = new ArrayList<Param>();
         switch(that.getValue()){ //controllo assegnamento
         case IFTT.Integer:
            int values = -1;
            if(that.getOp().contentEquals(IFTT.Assign))
               values = Integer.parseInt(that.getNameAssigned());
            else if(that.getOp().contentEquals(IFTT.MoreAssign))
               values = (int) (param.getValue() +
                   Integer.parseInt(that.getNameAssigned()));
            else if(that.getOp().contentEquals(IFTT.LessAssign))
               values = (int) (param.getValue() -
                   Integer.parseInt(that.getNameAssigned()));
            else if(that.getOp().contentEquals(IFTT.MulAssign))
               values = (int) (param.getValue() *
                   Integer.parseInt(that.getNameAssigned()));
            else if(that.getOp().contentEquals(IFTT.DivAssign))
               values = (int) (param.getValue() /
                   Integer.parseInt(that.getNameAssigned()));
            if(values != -1){
```

```
                param.setLastValue(param.getValue());
                param.setValue(values);
                list.add(param);
                SmarthomeCore.instance.setValues(list);
                return true;
            }
            return false;
        break;
        //ecc per i successivi casi....
    }
    return false;
}

public boolean visit(If ifCond){ // visita che controlla se la
    condizione e' verificata
      resValue.setValue(ifCond.getName());
      return comparator.exec(resValue.visite(ifCond.getValue()),
          visitPost(ifCond), ifCond.getOp());
    }
    return false;
}

public Pair<String,Object> visitPost(If ifCond){ // visita che
    controlla il tipo di assegnamento
    switch(ifCond.getType()){
    case IFTT.Value:
      resValue.setValue(ifCond.getNameAssigned());
      return resValue.visite(ifCond.getValue());
    }
      //ecc per i successivi casi....
    return null;
}
}
```

Esempio di una generica automazione letta dal parser:

```
// Se lo stato del parametro abat-jour è uguale allo stato OFF e
<if value="State" type="Param" name="abat-jour" op="Eq"
value="State" type="Value" name="OFF" op="And"

// Se è a casa l'utente Gianmarco
value="IsAtHome" type="User" name="Gianmarco" op="Eq"
value="Boolean" type="Value" name="True" />
```

```
// Allora assegna allo stato del parametro abat-jour lo stato ON
<do  value="State" type="Param" name="abat-jour"
op="Assign" value="State" type="Value" name="ON" />
```

4.9 Notifiche push

Per permettere al sistema di poter notificare al client Android messaggi di differenti tipi è necessario un meccanismo che abiliti l'invio di notifiche e non obblighi il client ad effettuare continue richieste. A questo proposito si utilizza un servizio fornito da Google chiamato Google Cloud Messaging o GCM che permette agli sviluppatori di spedire dati in modo bi-direzionale dai client ai server. All'interno del progetto sara' necessario creare un gestore di notifiche che si occupi di smistare le differenti notifiche a client diversi e che si assicuri dell'avvenuta ricezione. Sara' così possibile spedire notifiche a tutti gli utenti, ad un utente specifico ed essere sicuri che la notifica sia arrivata a destinazione.

4.10 Supervisionare i processi

All'interno del sistema è essenziale che tutti i processi dipendenti o semi-dipendenti tra loro non mettano a rischio la stabilita' del sistema stesso. E' necessario che ogni processo definisca i propri permessi , grazie ai quali il sistema è in grado di classificare la sua importanza. Uno dei processi che lavora del tutto in modo autonomo e parte all'avvio del sistema è WatchDog che si occupa di tenere sotto controllo tutti i processi avviati dal sistema e dall'utente. Il thread si occupa di analizzare l'uso delle memoria, di rilevare tutti i processi registrati ad un listener che abbiano generato errore ed inoltre monitora l'uso delle risorse da parte dei moduli caricati a run-time.

4.11 Caricamento della configurazione

L'intero sistema ha bisogno di un metodo semplice e rapido per caricare l'intera configurazione. Una strategia può essere quella di utilizzare dei parser gia' esistenti ed ottimizzati in grado di leggere e scrivere rapidamente su file. Il formato piu' usato in questi casi è l'XML ed in Java esiste un framework chiamato JAXB che mette a disposizione un set di API per semplificare l'accesso e la realizzazione di documenti XML attraverso applicazioni scritte direttamente in Java, con l'uso di annotazioni. Dalla versione 1.6, JAXB è

incluso in Java SE. Ogni file di configurazione o contente informazioni per l'utente sono in formato XML. E' necessario creare un processo che gestisca il caricamento della configurazione, della sua modifica e del suo salvataggio.

Capitolo 5

Implementazione del client Android

Per far si' che ogni utente sia in grado di utilizzare il sistema in modo semplice e veloce è necessario adottare alcuni accorgimenti. Ogni volta che verra' aperta una nuova schermata, sarà necessario mostrare all'utente tutte le possibili funzionalita' che quella schermata può fornire, tramite dei menu' dettagliati . Durante la prima installazione la schermata iniziale mostrata dovrà contenere un'interfaccia in grado di instaurare una connessione col sistema senza conoscere le necessarie informazioni tecniche per raggiungere il server (vedi figura 5.6). Questo sarà possibile tramite la sua ricerca all'interno della rete con dei pacchetti broadcast, contenti informazioni riguardo il client che si sta connettendo ed il suo indirizzo all'interno della rete locale. Dopo aver instaurato la prima connessione col sistema sara' necessario configurare rispettivamente:

1. **Le informazioni dell'utente**, quali nickname, nome SSID della rete wifi e possibilita' di attivazione delle notifiche. In modo del tutto trasparente queste informazioni verranno reperite dallo smartphone (se esistenti) così da ridurre al minimo la configurazione da parte dell'utente. Un pacchetto contenete queste informazioni, compresa l'IdKey per abilitare il client alla ricezione di notifiche push, verrà spedito al sistema una volta completati tutti i passaggi(vedi figura 5.1).

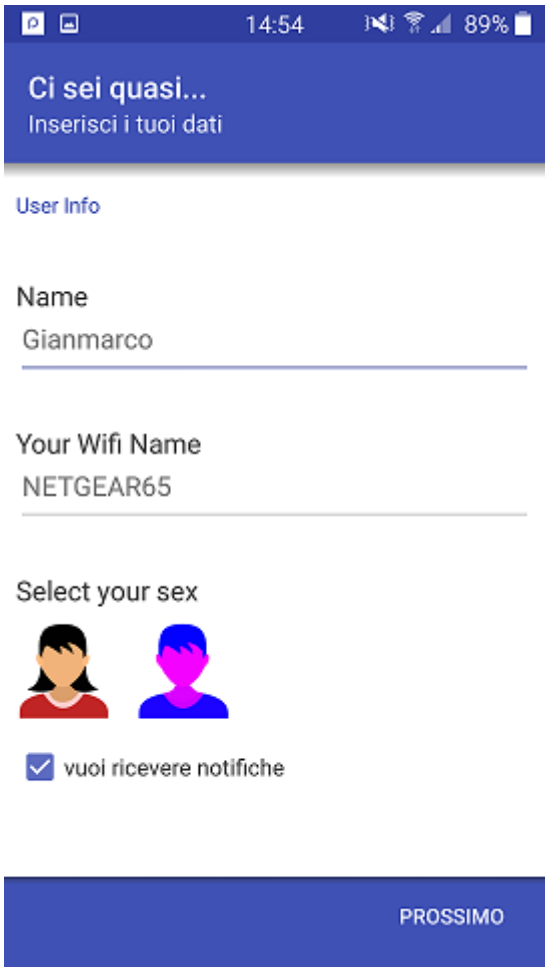

Figura 5.1: Configurazione delle informazioni sull'utente.

2. **Il tipo di contratto energetico utilizzato**. Sara' possibile selezionare il limite di consumi specificato dal proprio contratto con il prezzo per kWh dichiarato dalla societa' emittente del servizio. Nel caso in cui la tipologia di contratto non dovesse essere contenuto tra quelle disponibili, sarà possibile aggiungerne uno personalizzato. vedi figura 5.2).

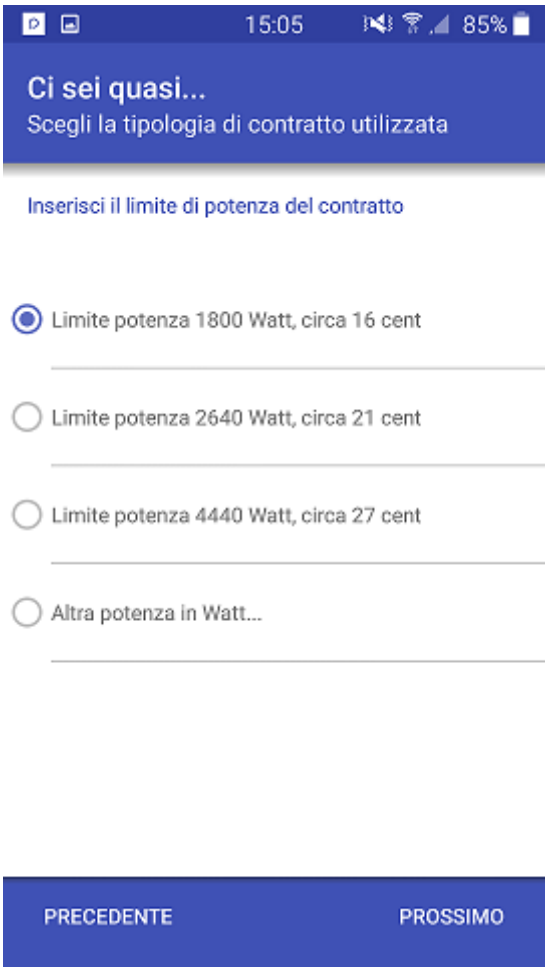

Figura 5.2: Configurazione del tipo di contratto.

3. **La configurazione dei dispositivi**. Per ogni elettrodomestico dichiarato nella configurazione del sistema, sara' necessario fornire un interfaccia grafica che permetta la configurazione in base alla tipologia del dispositivo, così da abilitare il sistema a processare correttamente i dati ricevuti da quel sensore. Ogni sensore verra' mostrato in base al tipo di utilizzo, quindi i layout potranno essere rappresentati dinamicamente al variare della tipologia. vedi figura 5.3).

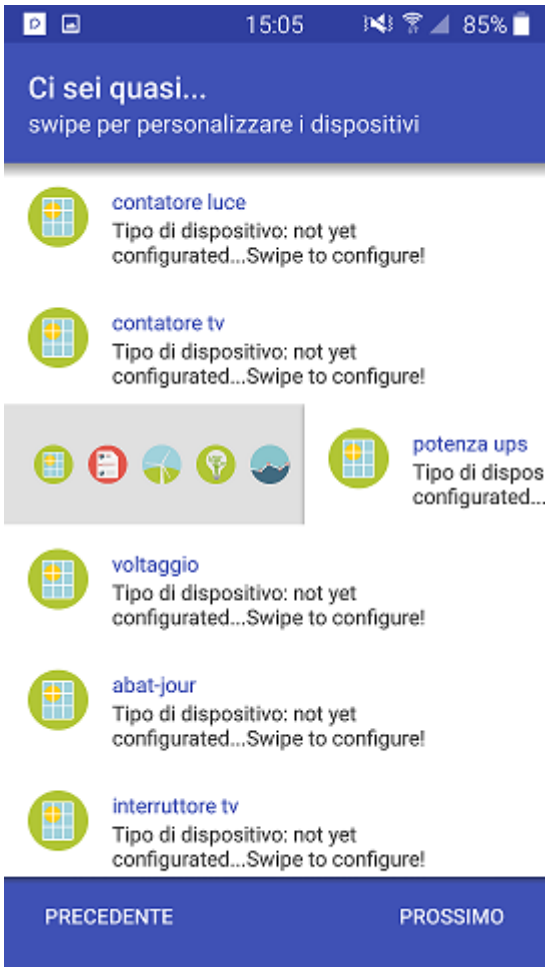

Figura 5.3: Configurazione dei dispositivi.

4. **L'ordinamento degli elettrodomestici da spegnere in caso di superamento della soglia**. Per evitare noiose interruzioni della corrente un menu' editabile mostrera' tutti i sensori configurati come interruttori, così da poter creare una priorita' di spegnimento al superamento dei consumi vedi figura 5.4).

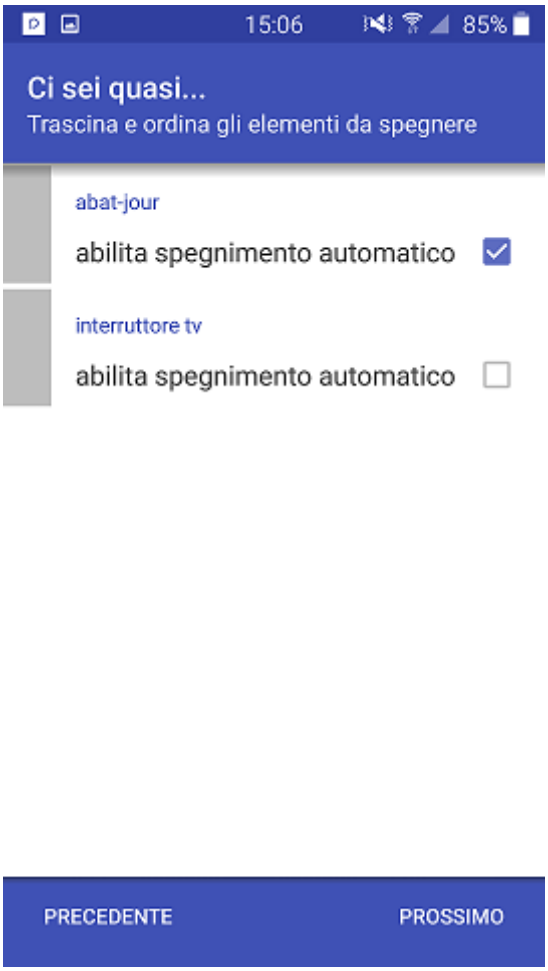

Figura 5.4: Selezione e ordinamento dei dispositivi da spegnere al superamento dei limiti imposti dal contratto.

5. **Gli elettrodomestici da spegnere all'uscita dell'utente**. Per ogni abitante della casa sara' possibile configurare uno o piu' dispositivi da spegnere nel caso in cui l'utente esca di casa. Due possibili scelte permetteranno all'utente di ricevere o una notifica riguardante la dimenticanza o lo spegnimento del tutto automatico del dispositivo (vedi figura 5.5).

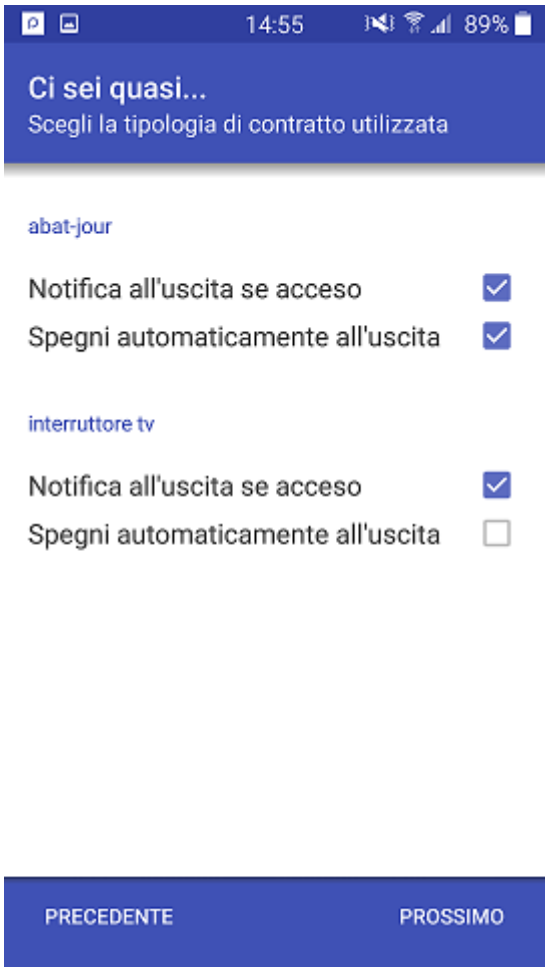

Figura 5.5: Selezione dei dispositivi da spegnere all'uscita dell'utente.

Figura 5.6: Schermata iniziale.

5.1 Gestione dei dispositivi

La gestione dei dispositivi collegati al sistema non richiede conoscenze specifiche. Infatti sara' possibile controllare i dispositivi in modo semplice tramite un menu' contenente tutti i dati inerenti al sistema (vedi figura 5.7). Ogni tipo di dispositivo verra' visualizzato coerentemente in base alla propria configurazione. Differenti layout saranno resi disponibile all'utente nel caso in cui si voglia modificare l'interfaccia di gestione (vedi figura 5.8).

L'aggiornamento dei dati mostrati su schermo avviene ogni circa 2 secondi con un continuo polling al sistema. Tramite il riconoscimento del tipo di device, sara' necessario creare differenti item della lista di gestione, così da distinguere dinamicamente ogni tipo di dispositivo configurato. Cliccando sui dettagli, verra' mostrata una finestra per la supervisione del dispositivo con informazioni inerenti ai consumi ed alle ore di accensione nell'arco mensile e giornaliero. Sara' possibile visualizzare il consumo del dispositivo selezionando una data specifica o quella attuale (vedi figura 5.9). Dopo aver scelto il periodo da visualizzare l'applicazione effettuera' una richiesta al sistema contenente la data ed il tipo di dispositivo, non appena vengono ricevute tutte le informazioni all'interno di un file Json verra' mostrato all'utente una schermata completa e dettagliata riguardante i consumi (vedi figura 5.10).

Figura 5.7: Schermata di gestione.

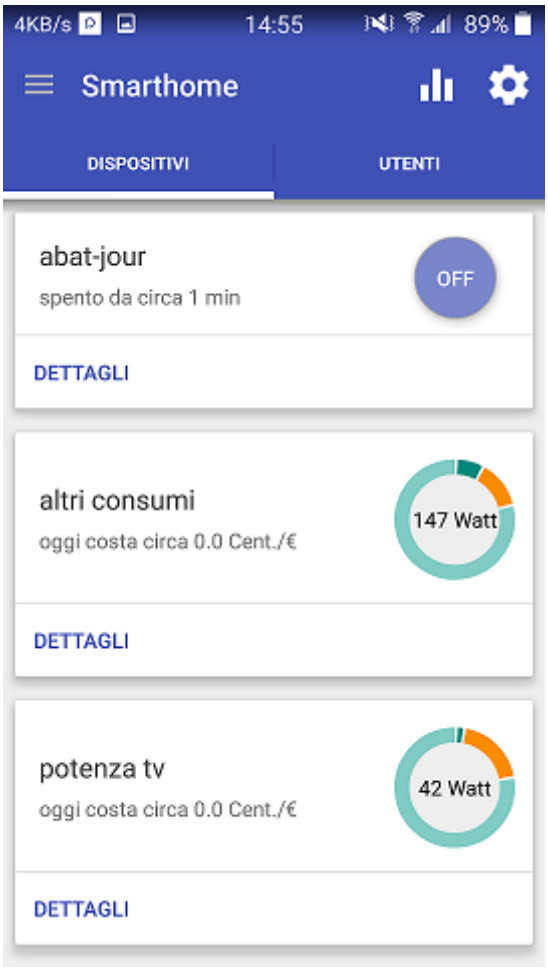

Figura 5.8: Visualizzazione di differenti tipologie di dispositivi (abat-jour visualizzata come interruttore, potenza tv visualizzata come contatore).

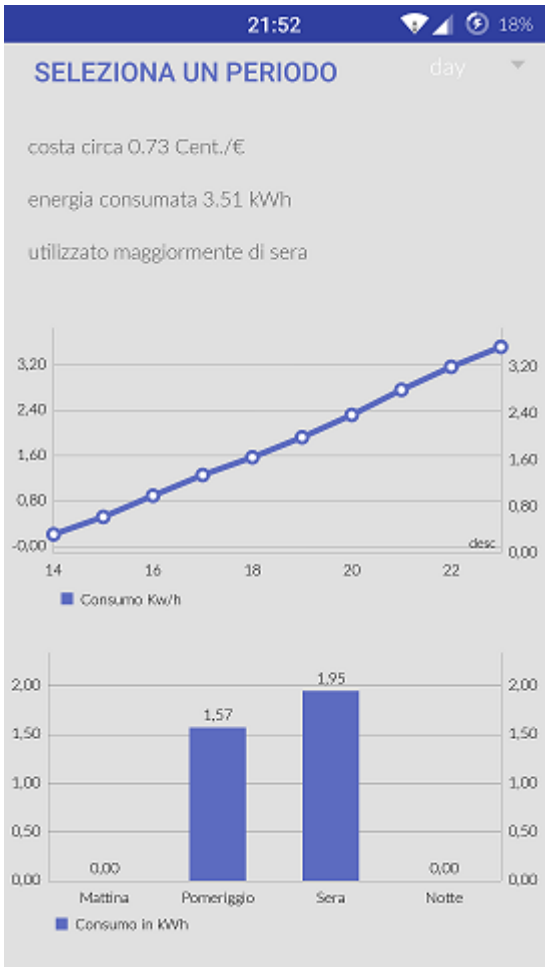

Figura 5.9: Visualizzazione del grafico dei consumi con la possibilita' di scegliere giorni/mesi differenti.

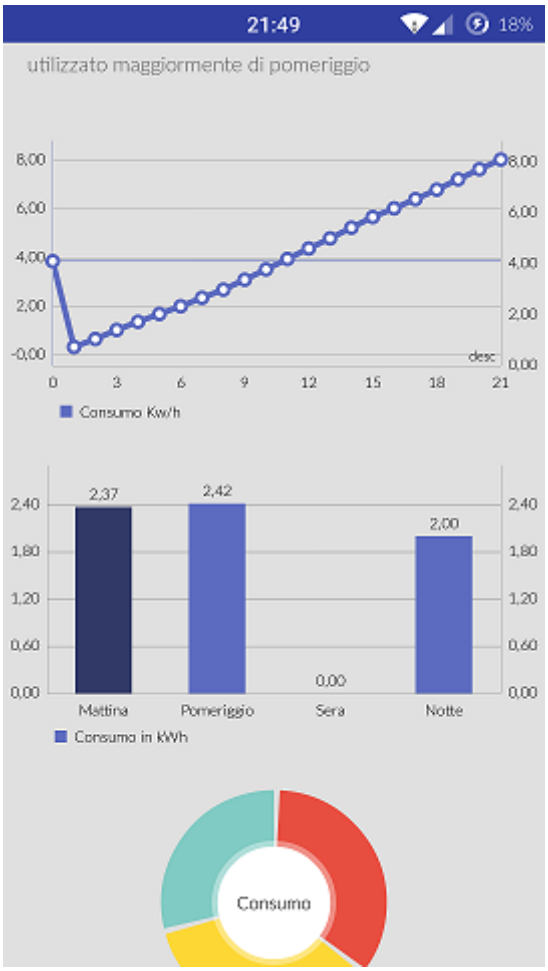

Figura 5.10: Grafico dettagliato sui consumi energetici e qualita' dell'uso del dispositivo.

5.2 Layout delle notifiche

Esistono differenti tipologie di notifiche, dalla semplice visualizzazione di uno stato del sistema alla gestione remota e automatica dei dispositivi. Ognuna di queste tipologie viene scelta dal sistema in base al tipo di evento da notificare. Tutte le tipologie sono contenute all'interno di un'interfaccia nell'SDK. Bastera' quindi analizzare il tipo di notifica ricevuta ed applicare il layout per la corretta visualizzazione.

Capitolo 6

Validazione sperimentale

L'intero sistema e' in grado di analizzare ed elaborare dei valori resi disponibili tramite i plugin. Ognuno di questi valori permette di avere una visione generale dell'andamento dei consumi dei dispositivi e della loro gestione. La correttezza dell'analisi è dipendente dalla qualita' dei valori ricevuti. Si intende l'accuratezza della provenienza dei dati in base allo stato attuale del dispositivo. Se la qualita' e' inferiore rispetto ad una soglia prestabilita il valore viene marcato come non corretto evitando possibili errori derivanti da valori obsoleti o da dispositivi non connessi mantenendo coerenza tra valori e stato del dispositivo(vedi figura 6.1).

ID	DATE	HOUR	POWER(W)	QUALITY	TODAY_ENERGY
0	2015-08-10	10:40:01.000000	299.0E0	180	9.402185430833336E0
0	2015-08-10	10:41:00.000000	302.0E0	180	9.406423776944445E0
0	2015-08-10	10:42:02.000000	336.0E0	180	9.41172460277778E0
0	2015-08-10	10:43:02.000000	303.0E0	180	9.417179227777781E0
0	2015-08-10	10:44:00.000000	341.0E0	180	9.422206263333333E0
0	2015-08-10	10:45:03.000000	344.0E0	180	9.428175063888888E0
0	2015-08-10	10:46:01.000000	301.0E0	180	9.441316786666668E0
0	2015-08-10	10:47:01.000000	2444.0E0	180	9.466548053611113E0
0	2015-08-10	10:48:00.000000	339.0E0	0	9.466548053611113E0
0	2015-08-10	10:49:02.000000	339.0E0	0	9.466548053611113E0
0	2015-08-10	10:50:03.000000	339.0E0	180	9.470761771944447E0
0	2015-08-12	16:31:00.000000	312.0E0	180	25.629284730000001E0
0	2015-08-12	16:32:02.000000	317.0E0	180	25.634606643333342E0
0	2015-08-12	16:33:01.000000	314.0E0	180	25.63967063138889E0
0	2015-08-12	16:34:01.000000	302.0E0	180	25.644866275E0
0	2015-08-12	16:35:00.000000	325.0E0	180	25.650014271666663E0
0	2015-08-12	16:36:03.000000	315.0E0	180	25.655695149444437E0
0	2015-08-12	16:37:02.000000	1342.0E0	180	25.674890632499995E0
0	2015-08-12	16:38:01.000000	1358.0E0	180	25.697030442222218E0

Figura 6.1: Snapshot del database e della correttezza del calcolo dell'energia. In rosso i valori delle energie che non crescono se la qualita' del dato non e' sufficiente. All'accensione successiva il dato e' di nuovo corretto, il sistema computa l'energia consumata dall'ultimo log.

6.1 Tempi di risposta e convergenze

Un importanza fondamentale nella gestione efficiente delle risorse del sistema e quella dell'uso di Timeout dinamici che permettano di adattarsi ai tempi di latenza dei dispositivi, senza bloccare in modo prolungato le risorse disponibili. Il metodo utilizzato e' quella del calcolo degli intervalli di confidenza su una coda contenente gli ultimi 10 tempi di risposta registrati. Viene definito a priori un limite massimo di attesa, normalmente piu' alto del previsto. Con l'aggiunta dei timeout salvati all'interno della coda, quelli generati tenderanno a convergere sempre piu' ad i primi evitando attese predefinite in caso di scarse comunicazioni. Se il timeout generato non dovesse bastare, ed il plugin non dovesse rispondere, viene aggiunto in coda il timeout massimale permettendo al valore di aumentare e di ridursi (vedi figura 6.2).

Plugin	Timeout	Timeout Generato	Data	Ora
SmarthomeArduinoDriver	206	215	2015-08-14	17:19:31.768
SmarthomeArduinoDriver	204	215	2015-08-14	17:19:57.739
SmarthomeArduinoDriver	202	216	2015-08-14	17:20:05.152
SmarthomeArduinoDriver	206	215	2015-08-14	17:20:12.569
SmarthomeArduinoDriver	204	214	2015-08-14	17:20:27.404
SmarthomeArduinoDriver	203	215	2015-08-14	17:20:38.527
SmarthomeArduinoDriver	297	4000	2015-08-14	17:20:46.042
SmarthomeArduinoDriver	202	241	2015-08-14	17:20:57.165
SmarthomeArduinoDriver	204	241	2015-08-14	17:21:04.582
SmarthomeArduinoDriver	204	241	2015-08-14	17:21:11.993
SmarthomeArduinoDriver	204	241	2015-08-14	17:21:23.116
SmarthomeArduinoDriver	203	213	2015-08-14	17:21:37.951
SmarthomeArduinoDriver	200	213	2015-08-14	17:21:45.361
SmarthomeArduinoDriver	202	213	2015-08-14	17:21:56.480
SmarthomeArduinoDriver	202	212	2015-08-14	17:22:03.898
SmarthomeArduinoDriver	205	215	2015-08-14	17:22:44.714
SmarthomeArduinoDriver	206	216	2015-08-14	17:22:55.844
SmarthomeArduinoDriver	205	217	2015-08-14	17:23:06.975
SmarthomeArduinoDriver	204	217	2015-08-14	17:23:14.389

Figura 6.2: Intervalli di confidenza. In rosso un attesa maggiore da parte del plugin. Il sistema aggiunge il limite massimo del timeout permettendo a quello generato di crescere(rappresentato verde) e riavvicinarsi al primo .

6.2 Risultati e correttezza delle operazioni

Ridurre i sensori sparsi per casa con un unico sensore collegato al contatore principale potrebbe ridurre i costi dell'acquisto dell'hardware, implementando la possibilita' di disaggregare le differenti tipologie di elettrodomestici in base al loro consumo. Nel dettaglio il sensore puo' misurare differenti variabili: corrente, voltaggio, potenza reattiva e potenza attiva. Avendo differenti valori si è in grado di distinguere gli elettrodomestici per esempio distinguendo lo scalda bagno ed il frigo confrontando i consumi della potenza reale e apparente. Lo scalda bagno e' un carico puramente resistivo. Il frigo invece è un carico maggiormente induttivo. Queste due variabili permettono di distinguere la maggior parte degli elettrodomestici costruendo un grafico a due dimensioni che rappresentera' le due differenti potenze. Questo potente strumento in grado di distinguere differenti dispositivi e' detto NILM (Non Invasive Load Monitoring) o monitoraggio del carico non invasivo, inventato da George Hart, Ed Kern e Fred Schweppe al MIT negli anni 80' (vedi figura 6.3). L'algoritmo originale consiste in 5 passaggi:

1. Un rilevatore di “bordi” che indentifica dei cambiamenti riconosciuti come possibili eventi di accensione/spegnimento.

2. Analisi dei cluster per individuare i cambiamenti tramite uno spazio bidimenzionale che indentifica la “firma” di un particolare elettrodomestico.

3. Cluster simili vengono accoppiati alla firma degli elettrodomestici. Questo passaggio riconoscere carichi semplici a due stati come accensione/-spegnimento ma non riesce ad associare pienamente cluster attribuibili a dispositivi complessi come la lavastoviglie che ha molti piu’ stati.

4. I gruppi non corrispondenti sono associati con cluster esistenti o nuovi, tramite un algoritmo di probabilità . Questo passo è noto come risoluzione anomalia.

5. Agli eventi vengono assegnate etichette leggibili (ad esempio “boiler” piuttosto che “carico 213”), facendo corrispondere gli eventi a consumi di elettrodomestici conosciuti, contenuti all’interno del database, appreso durante una fase iniziale di analisi.

Grazie all’uso di questo algoritmo sarebbe possibile ridurre al minimo i costi dell’hardware da installare all’interno dell’appartamento abbattendo i costi e la difficolta’ di installazione.

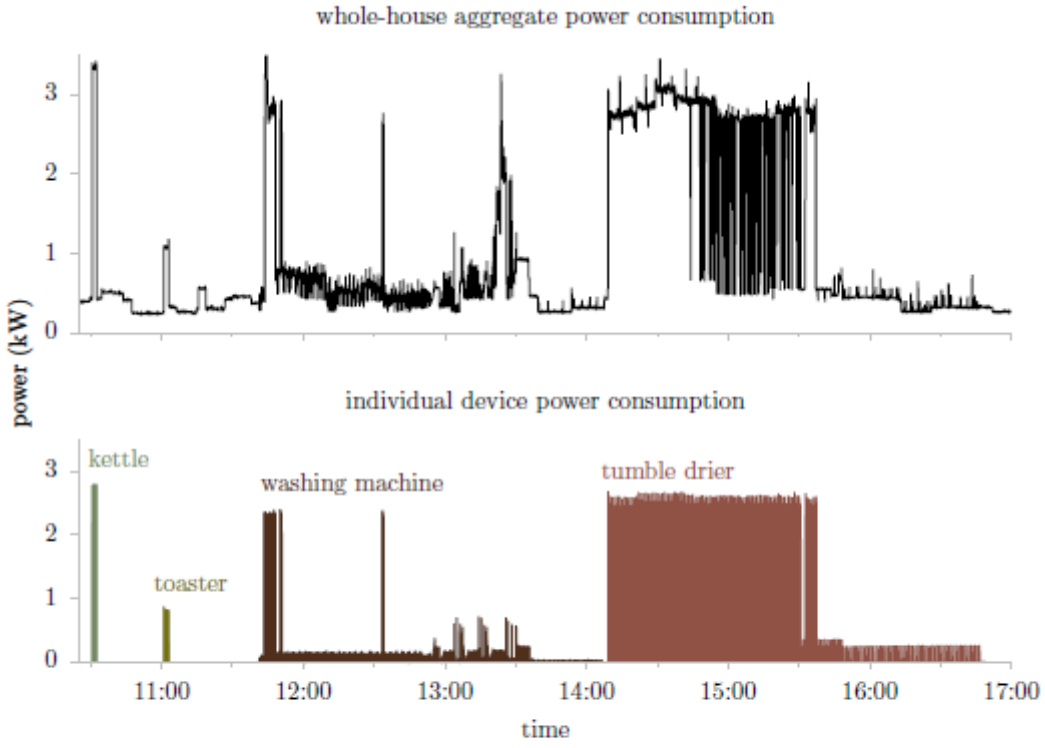

Figura 6.3: Esempio di disagreggazione della potenza tramite il tool NILM.

Capitolo 7

Conclusioni e Sviluppi futuri

Progettare e implementare un sistema domotico è un processo abbastanza complesso che richiede l'uso di hardware e software specifici. Il progetto descritto e' semplice nell'utilizzo e permette una gestione di base degli elettrodomestici. E' possibile estendere le sue funzionalita' tramite delle automazioni ed e' capace di interfacciarsi con ogni dispositivo tele-leggibile a patto che il protocollo sia stato aggiunto tramite forma di plugin all'interno del sistema. La sua flessibilita' pero' lo rende complesso durante l'installazione di nuove tipologie di elettrodomestici, nel caso in cui il protocollo richieda specifiche definite dall'utente. Il principio e' quello di mappare ogni scheda elettronica tele-leggibile con ogni elettrodomestico così che il sistema sia in grado di seguire un flusso di analisi in base alla tipologia di apparecchio.

7.1 Problematiche

Gestire gli elettrodomestici non e' semplice, ognuno può avere funzionalita' del tutto differenti da un altro e solo alcuni di questi si possono gestire tramite smartphone. Progettare dei microcontrollori in grado di integrarsi con le migliaia di elettrodomestici esistenti e' quasi impossibile per via delle modifiche invasive che dovrebbero essere apportate . E' possibile aggiungere manualmente sul sistema tutte le tipologie di dispositivi ma soltanto alcune verranno analizzate completamente. Il sistema si limita all'uso di interruttori per accendere/spegnere gli elettrodomestici e contatori per monitorare i consumi evitando modifiche sostanziali all'apparecchio. E' facile dedurre quindi che maggiori sono le funzionalita' che si hanno per il controllo di un elettrodomestico(nato senza il concetto di gestione remota) piu' invasiva sara' la tecnologia utilizzata per interagire con questo. Se si riuscisse a gestire in modo completo ogni elettrodomestico sarebbe possibile avere un incremen-

to notevole dei risparmi.Un ulteriore risparmio si potrebbe avere riducendo il numero delle schede elettroniche aumentando la complessita' del software riducendo quella dell'hardware con l'uso dell'algoritmo NILM (trattato nell'introduzione). Purtroppo ad oggi l'implementazione non sarebbe efficiente tanto quanto una supervisione specifica per ogni elettrodomestico. Resta quindi la necessita di hardware aggiuntivo per il controllo remoto.

7.2 Possibili estensioni

L'installazione del sistema potrebbe essere resa piu' semplice facendo uso delle due tipologie di schede elettroniche progettate. Cosi' facendo l'installazione sarebbe rapida e del tutto automatica; e' ovvio che nel caso in cui si voglia estendere il controllo bisognera' aggiungere dei controllori differenti ma in questo caso sara' possibile richiedere un plugin aggiuntivo allo sviluppatore, da rendere poi disponibile sul market dei plugin. E' necessario inoltre fornire all'utente un interfaccia web per la gestione tramite browser, si potrebbe quindi implementare una web app, capace di gestire e configurare l'intero sistema parallelamente all'applicazione android rendendo ancora piu' semplice l'uso del sistema.

Bibliografia

[1] Articolo - Il mercato della domotica in Italia: Stato e prospettive. *http://cetri-tires.org/press/2013/il-contributo-della-domotica-alla-gestione-energetica-e-sociale-delledificio-di-marco-sambati-e-pa?lang=it*

[2] Articolo - Il contributo della domotica. *http://www.cillario.net/index.php?option=com_content&view=article&id=3%3Ail-mercato-della-domotica-in-italia-stato-e-prospettive&catid=8&Itemid=109*

[3] Articolo Scientifico - MIT Power Disaggregation. *https://lids.mit.edu/research/research-highlights/power-disaggregation*

[4] Articolo - Tutto e' piu' smart con l'Internet of Things. *http://illuminotronica.it/una-casa-sempre-piu-smart-con-linternet-of-things/*

[5] Software - FreeDomotic: Open IoT development framework. *http://freedomotic.com/*

[6] Articolo Scientifico - Wireless Architectures for Heterogeneus Sensing in Smart Home Applications. *http://ieeexplore.ieee.org/xpl/articleDetails.jsp?reload=true&arnumber=6568872*

[7] Articolo Scientifico - Residential Appliance Indentification And Future Usage Prediction from Smart Meter. *http://ieeexplore.ieee.org/xpl/login.jsp?tp=&arnumber=6699944&url=http%3A%2F%2Fieeexplore.ieee.org%2Fiel7%2F6683943%2F6699103%2F06699944.pdf%3Farnumber%3D6699944*

[8] Soluzioni - Smart Domotics: confort e risparmio energetico. *http://www.smartdomotics.it/smart_dom_pro*

[9] Software - Movicon Industrial Automation Software. *http://www.progea.com/it-it/home.aspx*

[10] Immagine - Sistema di Domotica. *http://www.elenabiason.it/2015/07/09/passivhaus-la-bioarchitettura-e-luso-della-domotica*

[11] Articolo Scientifico - Imperial College London: Disaggregating Smart Meter Readings using Device Signatures. *http://www.doc.ic.ac.uk/teaching/distinguished-projects/2011/d.kelly.pdf*

[12] Articolo - Osservatorio sull'efficienza energetica 2013. *http://www.domotica.it/2013/09/osservatorio-sullefficienza-energetica-2013-seconda-edizione/*

[13] Articolo - The Importance of IPv6 and the Internet of Things. *https://securityintelligence.com/the-importance-of-ipv6-and-the-internet-of-things/*

[14] Specifiche Protocollo - MQTT Protocol Specification. *http://public.dhe.ibm.com/software/dw/webservices/ws-mqtt/mqtt-v3r1.html*