# Scontrol

## Gestione comandi

Spegnimento:

spegni l’elettrodomestico, valuta se si può riattivare un elettrodomestico, poi manda lo stato aggiornato all’utente

## Ricezione dati consumo.

Prima di aggiornare il dato sul consumo devi controllare se l’elettrodomestico era in accensione ed il consumo ricevuto è la metà (o se vuoi puoi fare solo con un minore) dell’ultimo consumo registrato. In tal caso l’elettrodomestico va in fase di esercizio, devi controllare se è necessario disattivare un qualche elettrodomestico, poi mandi lo stato aggiornato all’utente

## Cosa succede se scatta un timeout

Si spegne l’elettrodomestico e si manda lo stato nuovo a UserCmd. Attento: devi controllare se lo spegnimento dell’elettrodomestico può provacare una riattivazione

# Configurazione

Aggiungi nel file configurazione l’intervallo con cui i sensori inviano i dati di consumo e sistema di conseguenza Edi.java

# Test

Package elettrodomestico

Modificare i metodi equals

fatto perReportElettrodomestico, status, comando usercmd,DatiSensore ,rappresentazioneElettrodomestico (fatto tranne per le classi che usano contact ( Sensore, Scontrol e UserCmd), e per SafetyTimer, dove due timer non sono uguali se hanno gli stessi attributi

Package messaggi ( mancano ReportElettrodomestico e Status, che però hanno solo dei getters)

## Package usercmd

## Package scontrol

Interruttore

Timer

Scontrol

Comando Accensione con spegnimento preventivo

Comando Spegnimento con riattivazione

Arrivo dati sensore con entrata in esercizio

Arrivo dati sensore con entrata in esercizio e disattivazione

# Altro

Controlla gli intervalli di sicurezza e quelli di invio sensore: forse hai fatto un poì di confusione tra int e long

Fai i test per le classi report Elettrodomestico e Status del package messaggi, che sono fatte solo di getters.

Decidi se fare in modo che non si possa ordinare un’altra accensione se c’è un elettrodomestico in fase di avvio .

Vedi se c’è un modo di cancellare il countdown di un timer, anziché semplicemente disiscriversi all’evento.

Cancella tutta la parte relativa ai comandi di Scontrol a UserCmd nel apckage messaggi

# Versione senza contact

## Sensore

Nel metodo sendData, anziché mandare i dati tramite contact fai sì che ogni mezzo secondo venga invocato il metodo riceviDatiSensore SC, anziché mandare i dati tramite contact

## UserCmd

nel metodo MandaComando a Scontrol anziché il metodo userCmdDemand si deve invocare il metodo riceviEdElaboraComandoUserCmdFT di Scontrol e memorizzare lo status restituito

## Scontrol

Puoi commentare ciò che avveniva nei metodi Dojob e run del thread, visto cheScontrol non deve più attendere mesaggi contact provenienti da Sensori e UserCmd, ma riceve solo normali chiamate

## Riferimenti e configurazione

Sia i sensori che UserCmd devono avere un’attributo Scontrol, con relativo setter. Il riferimento dei sensori viene settato nel metodo configure della classe Edi, mentre quello di userCmd viene fatto all’internoo del metodo getInstance che ha come paramentro userCmd di Scontrol. Lasci comunque invariato in Edi l’assegnazione di nomi ai processi e il loro avvio.

# Interfaccia utente

UserCmd deve, tramite il pattern observer, notificare l’interfaccia utente quando il suo status cambia. Quindi UserCmd deve implementare i metodi ( alla observer) addGui, NorifyGui, Remove Gui. I primi due aggiungono o tolgono ad un vettore di gui un oggetto che implementa IGui. il terzo chiama i metodi update delle gui passando come parametro lo status .