Progetto Data flow

Progetto Data flow

In questo progetto raccoglieremo dati da un sensore collegato ad un Arduino. I dati raccolti verranno inviati tramite un modulo Bluetooth HC-05 al Raspberry Pi. Il Raspberry Pi riceverà i dati, li pubblicherà su un broker MQTT. Un altro software si sottoscrive a un topic MQTT per ricevere i dati, aggiornerà il suo stato e li renderà disponibili tramite un'API REST o simili.
I dati sono trasmessi in formato JSON con questa struttura: `{"type": "data_type", "value": value}`.
Ad esempio per la misura di una distanza potrebbe essere inviata nel formato: `{"type": "distance", "value": 3.5}`.

Obiettivi

Raccolta dati su arduino: Raccogliere periodicamente i dati da un sensore, es. HC-SR04, collegato ad un Arduino. Si può utilizzare un qualunque sensore l'obiettivo è solo quello di poter raccogliere dati.
Trasmettere i dati da Arduino a Raspberry Pi tramite un modulo Bluetooth HC-05.
Pubblicare i dati ricevuti su un broker MQTT.
Sottoscrivere un topic MQTT per ricevere i dati e aggiornare lo stato del sistema.
Esporre un'API REST per visualizzare i dati ricevuti.

Sviluppo

Si possono utilizzare le architetture di controllo che si ritengono più adatte. Il consiglio è di costruire un sistema ben architettato ma non eccessivamente complesso.
Libera scelta per i linguaggi e le librerie, l'importante è utilizzare i protocolli specificati sopra.
Per qualsiasi aspetto non specificato, fare la scelta che si ritiene più appropriata.

Risultati Attesi

data-collector (Arduino): Raccoglie i dati dal sensore di distanza HC-05 e li invia in formato JSON tramite Bluetooth.
data-publisher (Raspberry): Riceve i dati dal modulo Bluetooth HC-05, li decodifica e li pubblica su un topic MQTT.
data-subscriber: Si sottoscrive al topic MQTT, aggiorna il suo stato con i dati ricevuti e fornisce un'API REST per accedere ai dati.

Analisi e sviluppo arduino

Come base per il progetto siamo partiti dall'esercizio bt-comm fornitoci per creare una connessione bluetooth tramite il sensore HC-05.

Come sensore di raccolta dati abbiamo utilizzato il *Sensore ad ultrasuoni* per prendere ed inviare dati tramite bluetooth al raspberry. Per utilizzarlo abbiamo dovuto modificare il codice sorgente della **ProducerTask**, dandogli come argomento anche il puntatore al *Sonar*. Per semplificare l'invio dei dati tra dispositivi abbiamo optato per formattare il messaggio inviato al raspberry in formato Json, così da poterlo modificare liberamente in seguito.

Analisi e sviluppo raspberry

In questa fase ci doabbiamo occupare della recezione dati da parte dell'arduino e della successiva pubblicazione su un broker MQTT.

Abbiamo creato uno script unico che comprende la connessione via bluetooth e il caricamento dei dati su MQTT. Come prima cosa abbiamo modificato in buona parte lo script dato come esempio presente nella cartella *bt-comm*, così da avere una base per la comunicazione bluetooth. Una volta acccertata la connessione con l'arduino abbiamo gestito il flusso di dati in entrata e li abbiamo formattati in oggetti Json, così da poterli inviare a un broker MQTT.

HiveMQ

Inizialmente il prototipo funzionava solo in localhost e, una volta finito il testing e accertati del funzionamento siamo passati ad utilizzare il servizio ***HiveMQ***, creando un cluster e connettendoci gestendo la connessione *tls* al broker. Per fare questo ci siamo basati principalmente sulla guida per la connessione in python reperibile nella documentazione di HiveMQ.

Seguendo la documentazione abbiamo anche creato degli utenti con privilegi diversi per effettuare le varie azioni di *publish e subscribe*. Qui riportati nomi e password dei due utenti creati nel pannello degli accessi di HiveMQ:

Nome	Password
rootUser	rootPass1
subscriber	subPass1

Visualizzazione dati

Una volta accertati del funzionamento abbiamo pensato di creare una piccola pagina prototipo per mostrare i dati in modo *chiaro e più leggibile* di una semplice stringa su una shell, per questo con un po' di ricerca ci siamo imbattuti in streamlit, una libreria Python che permette di prototipare una pagina web in modo facile e veloce. Inoltre questa libreria supporta moltissimi widget per fare *plotting* di grafici, tabelle e tutto quello che riguarda i dati. La sincornizzazione della pagina è stata una bella sfida, dovendo controllare due theread, ovvero quello di *MQTT* e quello di *Streamlit*, ma alla fine abbiamo

fatto due sezioni, una con una tabella e una con un *line chart* per visualizzazre i dati. Per avviare la pagina streamlit bisogna aprire una shell nella cartella `/src/streamlit` e con il comando `streamlit run page.py` verrà avviata la pagina in locale.