

# WoT Store:

una piattaforma per l'interoperabilità  
semantica in contesti IoT e WoT

---

**Relatore:** Prof. Marco Di Felice

**Correlatore:** Dott. Luca Sciullo

**Candidato**  
Lorenzo Gigli

{ Università di Bologna  
Dipartimento di Informatica - Scienza e Ingegneria  
Corso di Laurea Magistrale in Informatica }

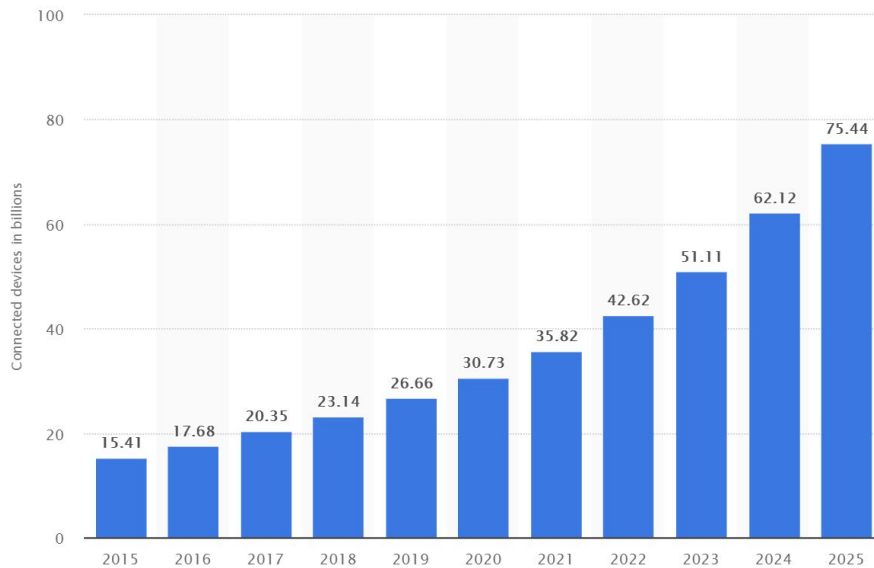
Stato dell'Arte

# Internet of Things (IoT)

Estensione di Internet al mondo degli oggetti e dei luoghi concreti.

Da semplici oggetti muniti di codice QR, NFC/RFID Tag a sistemi complessi come Smart Car, Smart Building, ecc.

Difficile interoperabilità per utilizzo di protocolli e formati diversi (basso livello) tra i numerosissimi prodotti esistenti.



**Numero di dispositivi connessi a livello mondiale dal 2015 al 2025**

<https://www.statista.com/statistics/471264/iot-number-of-connected-devices-worldwide/>

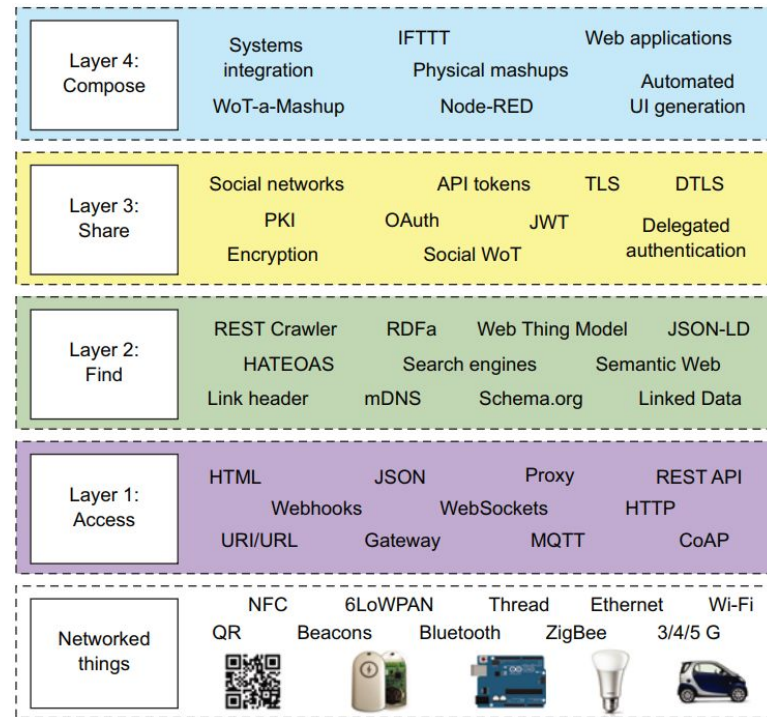
# Web of Things (WoT)

Utilizzo di standard e tecnologie web, che virtualizzando il concetto di Thing, mira a risolvere il problema di interoperabilità dell'IoT.

Si posiziona come espansione dello strato Application del modello OSI.

Possiede quattro livelli di funzionalità con gradi di integrazione e accessibilità via via più elevati.

Anche in questo caso esiste il rischio di una proliferazione di standard.



# W3C WoT

Un Working Group istituito dal W3C per contrastare la frammentazione dell'IoT attraverso componenti standard complementari:

- **WoT Thing Description (TD)**

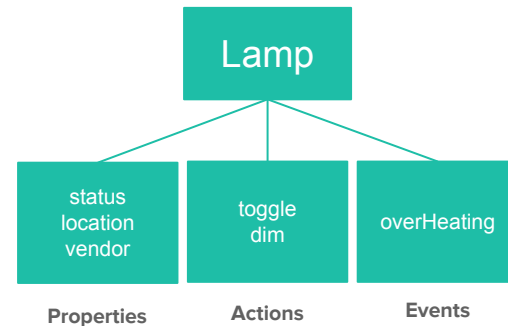
L'entry point di una Thing: definisce delle ontologie per la **descrizione semantica** dei dati, modelli di interazione (Properties, Actions, Events), meccanismi di sicurezza e pattern di comunicazione; la sua serializzazione standard è **JSON-LD**

- **WoT Binding Templates**

Collezioni di metadati per l'interoperabilità tra piattaforme IoT

- **WoT Scripting API**

API per l'interazione T2T e la gestione del lifecycle della Thing



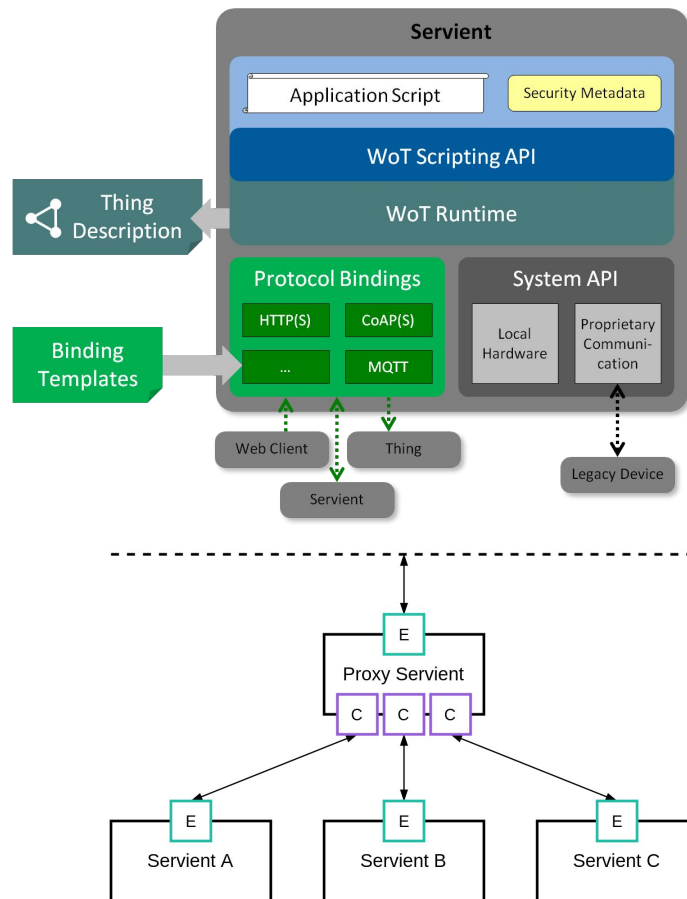
Ogni oggetto fisico o virtuale  
che abbia una TD associata  
è considerato una Thing

# W3C WoT: Servient

Un Servient è uno **stack software** che implementa i componenti del W3C WoT.

I Servient possono fungere da host per le Thing, **esporle** e/o **consumarle**, svolgendo così contemporaneamente sia la funzione di server che di client.

L'architettura W3C si basa quindi sulla costruzione di catene di astrazioni che forniscono un alto livello di interoperabilità.



# WoT Store

& Thing CLI

# Obiettivi e funzionalità

Obiettivo primario: realizzare una **piattaforma** per la **ricerca semantica** di applicazioni per il W3C WoT (Thing Application e Mashup Application), operandone **l'installazione** su Thing **compatibili** o la diretta esecuzione.

**Il sistema è confrontabile agli store di applicazioni mobile, ma per IoT.**

Funzionalità principali:

- Ricerca semantica di applicazioni e Thing
- Installazione/esecuzione automatica delle applicazioni
- Rendering delle Thing basato sul descrittore semantico (TD)



# Thing Application (TA) & Mashup Application (MA)

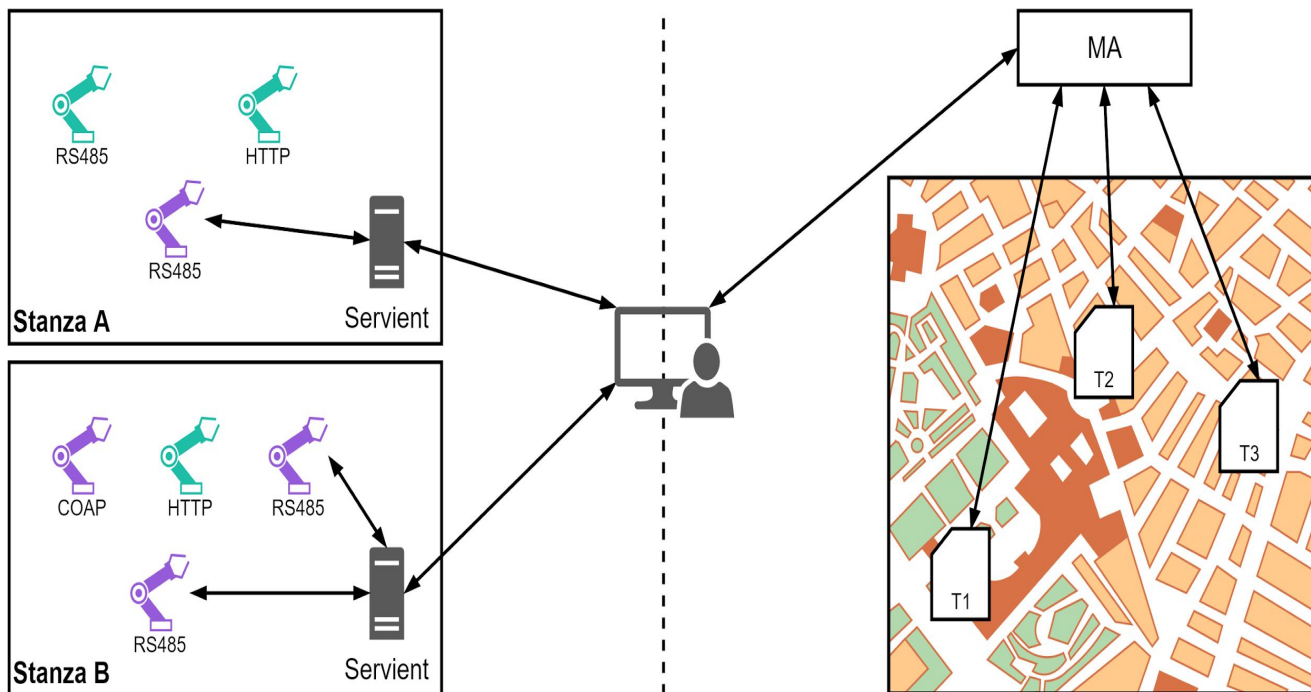
**Thing Application:** forniscono un'implementazione del comportamento di una Thing coerente con la sua TD.

Es. criteri di ricerca: tipo semantico della Thing, permessi richiesti dall'applicazione (root, GPIO, access, ecc.), piattaforma coinvolta (Arduino, Raspberry Pi, ecc.) e altro.

**Mashup Application:** si rapportano con un insieme di Thing, producendo nuovi output o fornendo nuovi servizi.

Es. criteri di ricerca: formato desiderato per la temperatura (Celsius), tipo di database (NoSQL), intervallo di aggregazione (ogni minuto).

# Casi d'uso



Aggiorna tutti gli attuatori **viola** che parlano **RS485**

Esegui una MA per le previsioni meteo di una certa zona

# Architettura

## Market Interface (MI)

Applicazione web che presenta l'interfaccia primaria dello store

## Thing CLI

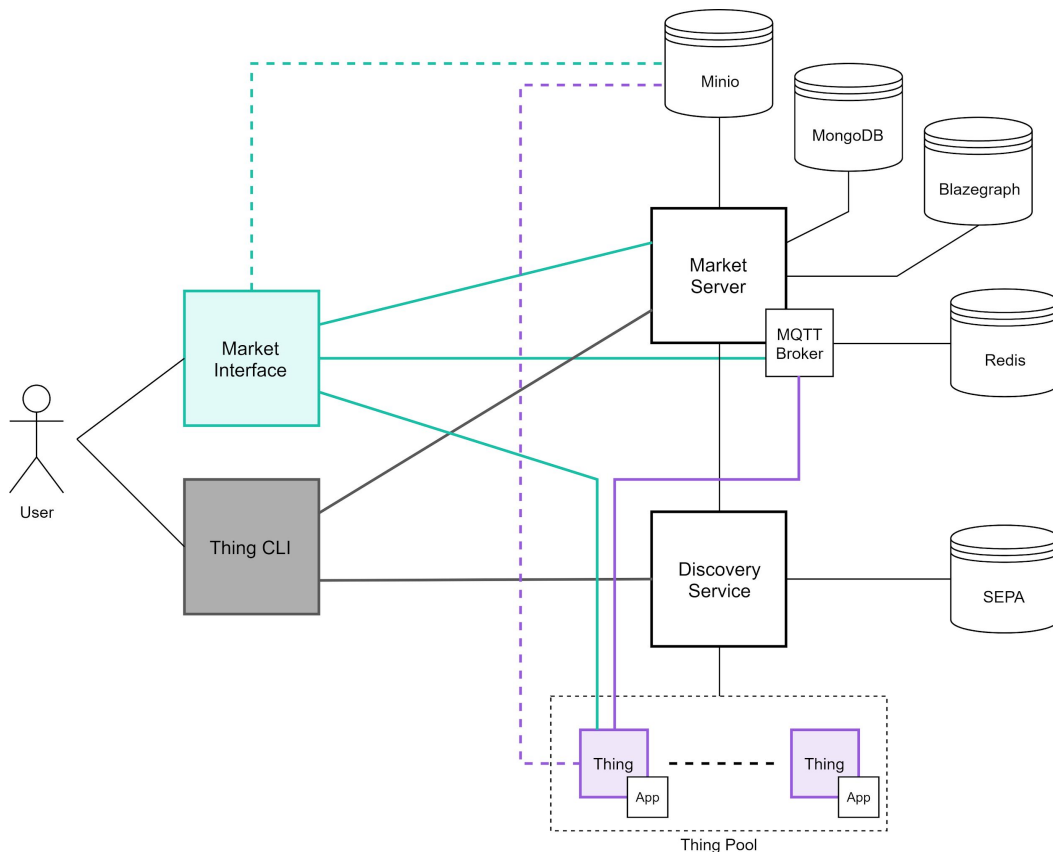
Strumento a riga di comando per installazione e configurazione del Servient, configurazione e registrazione delle Thing

## Market Server (MS)

API REST, WebSocket e broker MQTT

## Discovery Service (DS)

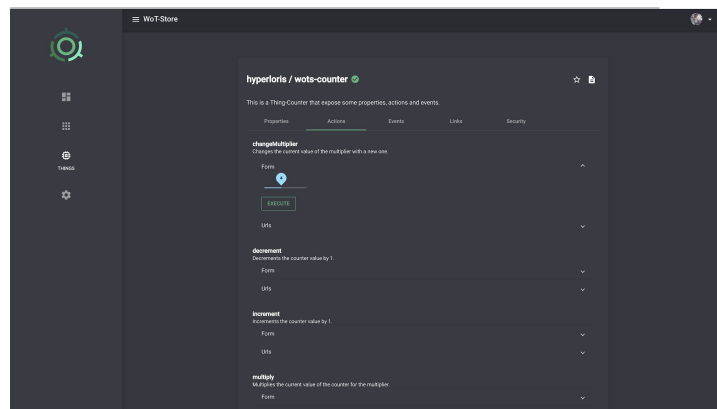
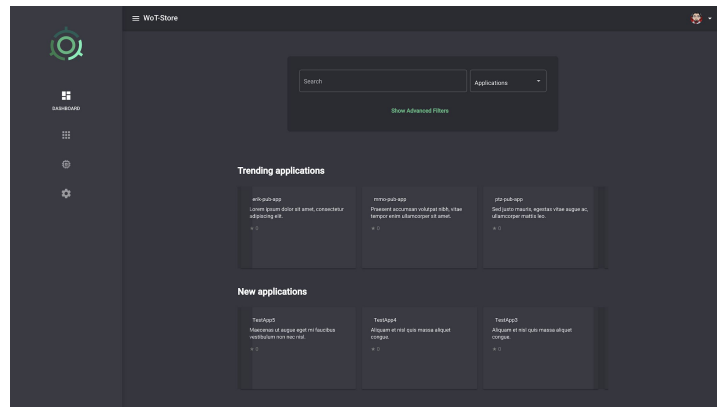
Monitoraggio dello stato delle Thing con notifiche dei cambiamenti (pub-sub)



# Market Interface

L'interfaccia del WoT Store è un'applicazione **Angular (v6)** altamente modulare che fornisce le seguenti funzionalità:

- **Ricerca** di utenti, applicazioni e Thing
- **Suggerimenti** automatici su applicazioni e Thing
- **Operazioni** CRUD sulle applicazioni e Thing  
La TD della Thing viene processata completamente e i suoi dati sono caricati **live** tramite vari protocolli; ne viene gestito anche lo schema di sicurezza.  
{ sistema di factory }
- **Installazione** TA sulle Thing, deploy MA su WoT Store
- **Condivisione** (sicura) di applicazioni e Thing  
Tramite sistema di ACL e ruoli utente.



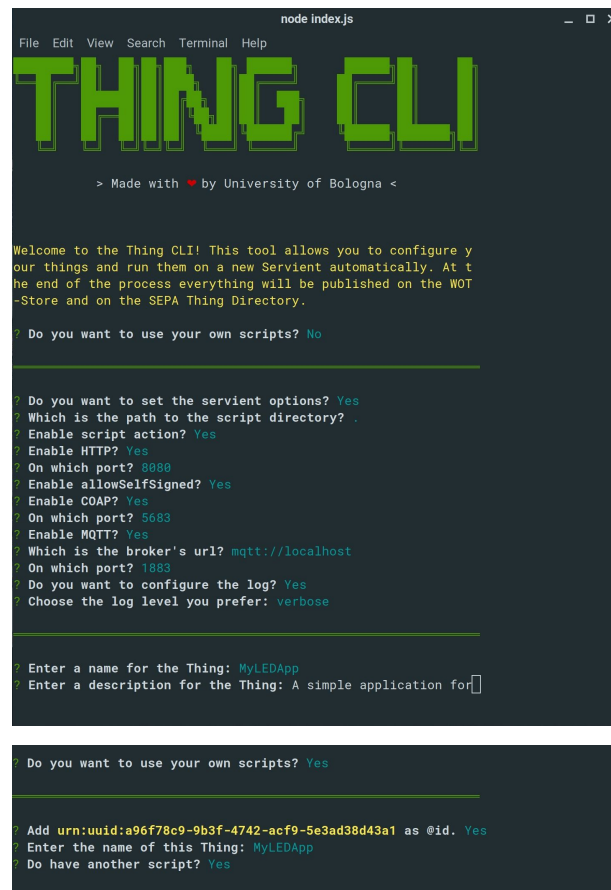
# Thing CLI

Si tratta di un'applicazione per terminale eseguibile su Windows, Linux e OS X. La portabilità è garantita dalla combinazione **Node.js + ShellJS**.

Permette all'utente di configurare e pubblicare facilmente e automaticamente le proprie Thing sul WoT Store e sul DS.

Fornisce le seguenti funzionalità:

- **Generazione** file di configurazione del Servient (o default)
- **Creazione** multi Thing da zero con applicazione default
- **Caricamento** multi Thing a partire dai file utente
- **Inizializzazione** e **avvio** del Servient
- **Registrazione** delle Thing su MS e DS



```
node index.js
File Edit View Search Terminal Help
THING CLI
> Made with ❤ by University of Bologna <

Welcome to the Thing CLI! This tool allows you to configure y
our things and run them on a new Servient automatically. At t
he end of the process everything will be published on the WOT
-Store and on the SEPA Thing Directory.

? Do you want to use your own scripts? No

? Do you want to set the servient options? Yes
? Which is the path to the script directory? .
  Enable script action? Yes
  Enable HTTP? Yes
  On which port? 8080
  Enable allowSelfSigned? Yes
  Enable COAP? Yes
  On which port? 5683
  Enable MQTT? Yes
  Which is the broker's url? mqtt://localhost
  On which port? 1883
  Do you want to configure the log? Yes
  Choose the log level you prefer: verbose

? Enter a name for the Thing: MyLEDAApp
? Enter a description for the Thing: A simple application for

? Do you want to use your own scripts? Yes

? Add urn:uuid:a96f78c9-9b3f-4742-acf9-5e3ad38d43a1 as @id. Yes
? Enter the name of this Thing: MyLEDAApp
? Do have another script? Yes
```

# Market Server



Il server dello store è un'applicazione **Node.js** (v10) che utilizza il framework **LoopBack** (v3) in combinazione con Socket.IO e un broker MQTT realizzato con la libreria Mosca.

Espone un'**API REST** per la gestione delle risorse, un **endpoint WS** per aggiornamenti real-time, un **endpoint MQTT** che fa da tramite tra gli eventi generati dalle Thing e il front-end Angular.

I dati vengono salvati su 4 database/storage differenti:

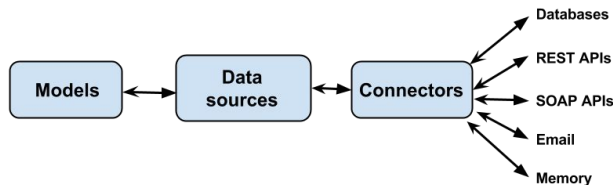
- **Minio**: object storage per i sorgenti delle applicazioni
- **MongoDB**: database per la gestione ordinaria degli utenti e delle relazioni tra i modelli
- **Blazegraph**: triplestore per salvare la TD delle applicazioni e delle Thing
- **Redis**: database key-value per il broker MQTT

## Connettore LoopBack per Blazegraph:

Trasformazione filtri in query SPARQL e conversione formati dati.

Server → Triplestore | JSON-LD → N-Triples

Triplestore → Server | JSON → JSON-LD



# Configurazione delle WSN

## BLE Network

3x NodeMCU BLE

1x Raspberry Pi Ethernet

I nodi comunicano direttamente con la System API\*.

## Wi-Fi Network

3x NodeMCU Wi-Fi su LAN 2

1x Raspberry Pi Wi-Fi su LAN 2 + LAN 1

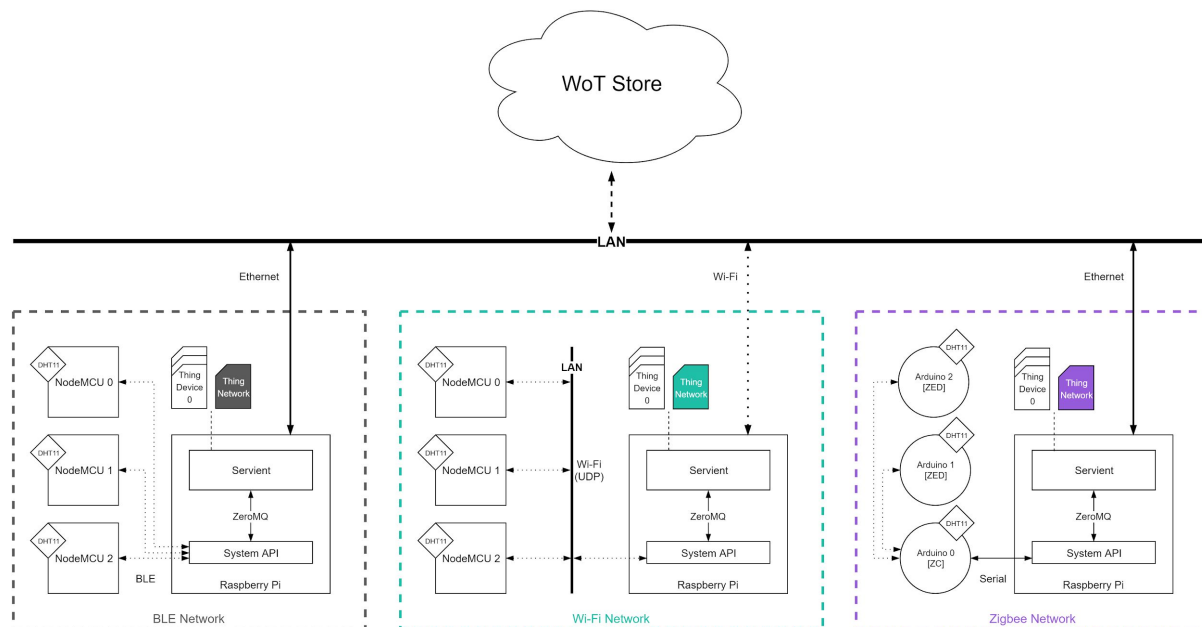
La System API ascolta i messaggi mandati in broadcast dai nodi sulla LAN secondaria

## Zigbee Network

3x Arduino con modulo Xbee

1x Raspberry Pi Ethernet

Il ZC comunica tramite seriale con la System API.



**\*System API:** il livello Device Query gestisce la comunicazione request-response con gli end-node, IPC rende i dati disponibili alla Scripting API del Servient.

# Esecuzione

Per mezzo del **Thing CLI** sono stati generati gli script di default con i metadati specifici per ogni nodo, inizializzati e configurati i Servient, registrate le Thing.

Quindi, tramite il **WoT Store**, sono stati installati due tipi di **TA** e lanciato il deploy di una **MA**:

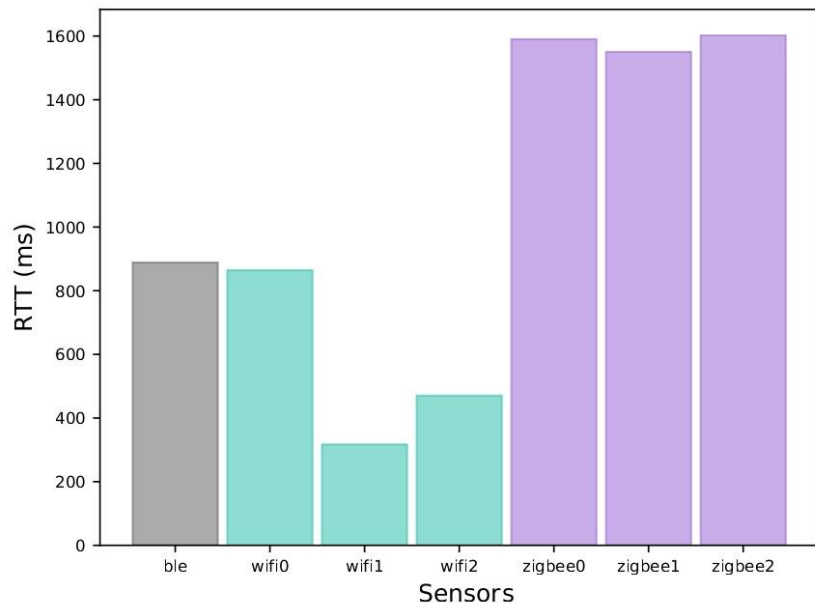
- **Thing Device** (uno per ogni end-node), che espone le proprietà, le azioni e gli eventi

Es. state, frequency, getData, setFrequency, newData, newState

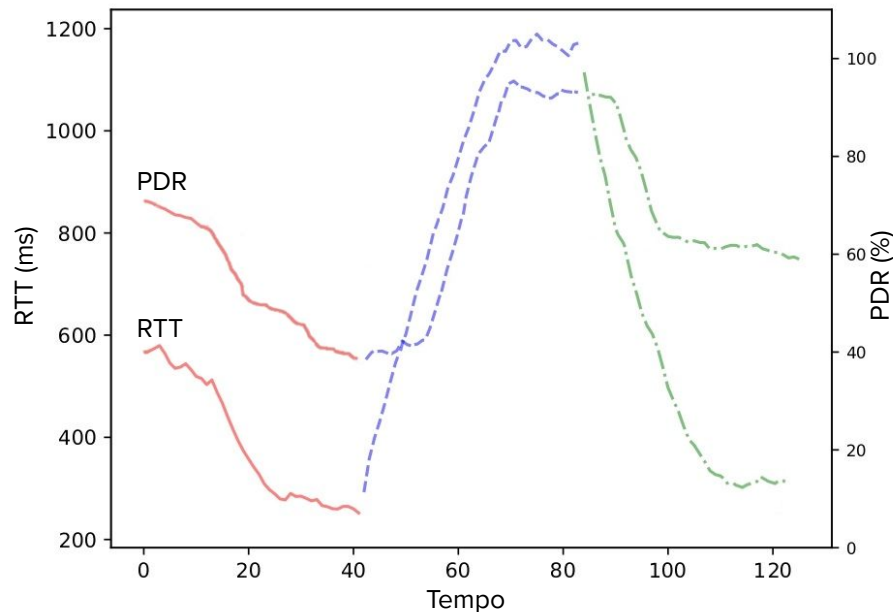
- **Thing Network** (uno per ogni Servient), che descrive la performance della propria WS
- **Mashup Application**, che consuma tutte le Thing, raccoglie i dati e produce un file CSV



# Risultati



Differenze tra i sensori in termini di RTT



Sostituzioni dinamiche della MA con cambio policy

1. Minimizza RTT
2. Massimizza PDR
3. Bilancia RTT e PDR

# Conclusioni

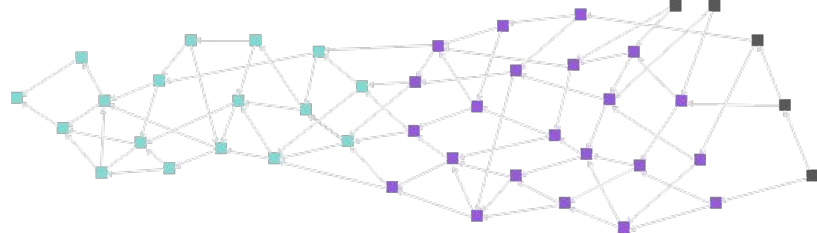
**WoT Store** gestisce applicazioni e Thing per il W3C WoT, con le seguenti principali peculiarità:

- Motore di ricerca **semantico**
- **Installazione** delle applicazioni
- **Visualizzazione** delle Thing

**Thing CLI** semplifica e velocizza l'inizializzazione di un nuovo Servient e delle Thing.

La sperimentazione condotta ha effettivamente dimostrato la validità e l'efficacia di questi strumenti nella **gestione automatizzata** di un sistema di reti di sensori eterogenee, indipendentemente dai dettagli tecnologici e dalla struttura fisica della rete.

# Sviluppi futuri



- Mantenere il lavoro **allineato** con lo standard W3C WoT
- Introdurre un sistema di **cron-job** per poter pianificare modifiche sulle Thing  
{Tecniche di Machine Learning per automatizzare e ottimizzare la pianificazione}
- Aggiungere la possibilità di **vendere le proprie applicazioni**
- Implementare la compravendita dei dati delle Thing tramite **microtransazioni automatiche T2T**
- **Da questa tesi è in corso di stesura un articolo di ricerca**

Grazie per l'attenzione

---