UNIVERSITÀ DEGLI STUDI DI PARMA
DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE
CORSO DI LAUREA IN INGEGNERIA INFORMATICA

ARCHITETTURE CLOUD A SUPPORTO DELLA COMUNICAZIONE PER SCENARI DI INTERNET OF THINGS

CLOUD ARCHITECTURES FOR COMMUNICATION SUPPORT
IN INTERNET OF THINGS SCENARIOS

Relatore:
Chiar.mo Prof. MARCO PICONE

Correlatore:
Dott. Ing. SIMONE CIRANI

Tesi di Laurea di:
MIRCO ROSA

Fra tutte le evoluzioni che stanno interessando il mondo dell'Informatica in questi tempi quella relativa all'Internet of Things è sicuramente una delle più importanti e profonde: promette di modificare drasticamente la società ed il modo in cui viviamo, introducendo metodi di interazione e scambio di informazioni fra persone e cose mai visti prima; indirizzare lo sforzo della ricerca in questo ambito è quindi particolarmente auspicabile e l'esponenziale progresso tecnologico registrato negli ultimi anni rende facilmente accessibili strumenti potenti adatti alla causa, sia in termini hardware che software.
Per questi motivi ci si è posti come obiettivo lo sviluppo di un'architettura software completa in grado di rendere fruibili informazioni di reti IoT locali ad utenti remoti connessi ad Internet, utilizzando protocolli prettamente dedicati all'Internet delle Cose e sfruttando le moderne tecnologie offerte dal Cloud in ambito di Computing, Storage e Messaging. Tra i molteplici fattori da tenere in considerazione in fase di progettazione abbiamo le prestazioni: l'obiettivo non sarà quindi soltanto sviluppare un sistema efficace e funzionalmente completo, ma operare scelte architetturali che permettano il mantenimento delle performance in scenari caratterizzati da grandi quantità di nodi e client.

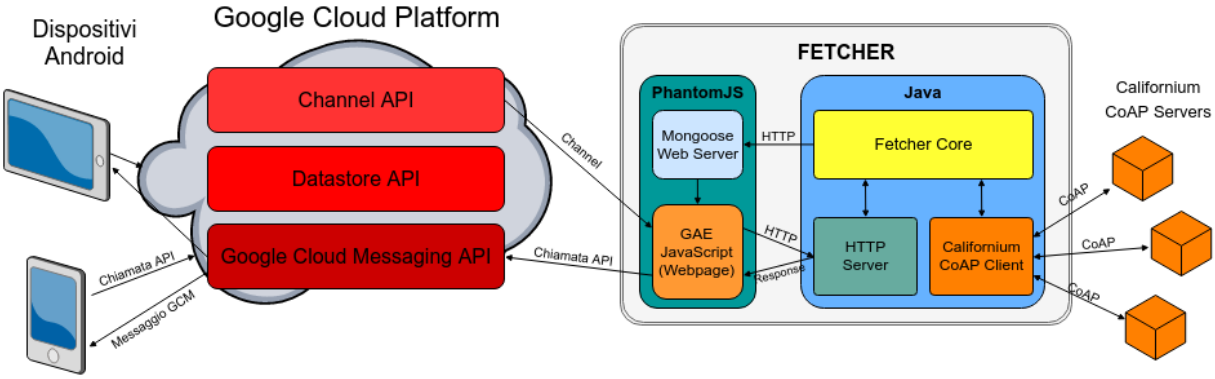
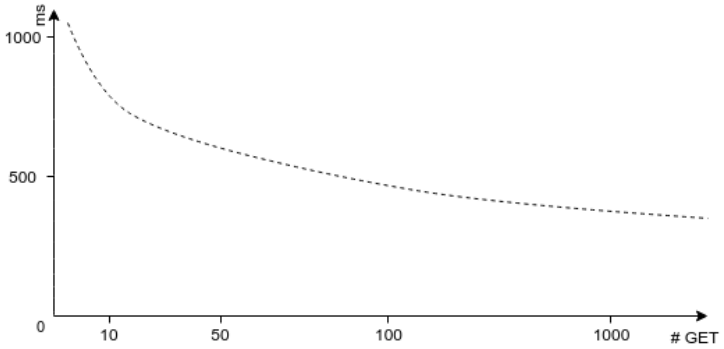


Figura 1: Architettura Completa

In figura 1 è rappresentata l'intera architettura sviluppata. E' stato scelto di implementare un nodo intermedio (il Fetcher) con il compito di gestire sia la rete locale IoT, utilizzando il framework CoAP Californium e la libreria di Service Discovery JmDNS, sia le comunicazioni con il Cloud avvalendosi dell'headless browser PhantomJS. Vi è poi la parte della Google Cloud Platform,

che provvede a memorizzare ed elaborare le informazioni provenienti dal Fetcher, inviare notifiche ai client Android tramite Cloud Messaging e reindirizzare le richieste che gli stessi Client rivolgono alle risorse IoT.
Il sistema così strutturato permette al Fetcher, utilizzando la Service Discovery, di rilevare in tempo reale la situazione (dinamica) dei nodi presenti nella LAN, sincronizzandosi costantemente con il Cloud; dall'App Android sarà quindi possibile ottenere la struttura dello scenario IoT ed effettuare richieste GET e POST alle singole risorse, nonché creare relazioni di observing in modo da ricevere automaticamente aggiornamenti real-time dai nodi interessati.
Dopo aver implementato e provato praticamente l'architettura possiamo affermare di aver concepito un sistema solido, prestante e flessibile: i nodi CoAP e l'App Android, che si trovano agli estremi della struttura e costituiscono il punto di contatto con il mondo reale, possono essere largamente personalizzati in base alle proprie esigenze, potendo contare su una robusta struttura interna in grado di adattarsi automaticamente a variazioni e modifiche di progetto. Le scelte implementative fatte consentono inoltre una buona scalabilità, mantenendo (e per certi versi aumentando) le prestazioni in scenari contraddistinti da un alto numero di nodi e client. Per mettere alla prova il sistema è stato effettuato un test sul tempo di risposta delle richieste GET all'aumentare della "frequenza oraria" di queste ultime:



Numero GET	Latenza Media
1-10	$\sim$ 1000ms
10-50	$\sim$ 680ms
50-100	$\sim$ 570ms
100-200	$\sim$ 520ms
200-300	$\sim$ 490ms
300-500	$\sim$ 470ms
500-1000	$\sim$ 440ms

Figura 2: Tabella e diagramma delle prestazioni

Come possiamo vedere in figura 2, inizialmente si ha una latenza relativamente alta, che però diminuisce costantemente all'aumentare del numero delle richieste: questo perché la Google Cloud Platform è progettata per gestire il carico di lavoro in modo intelligente, privilegiando ed offrendo tempi di risposta migliori alle istanze delle applicazioni con più traffico in entrata e uscita. Avremo quindi prestazioni migliori sulla piattaforma Cloud all'aumentare delle richieste da parte dei client Android, e la natura stessa del protocollo CoAP garantirà l'assenza di colli di bottiglia ed ottime performance della parte IoT anche in presenza di un numero di operazioni molto elevato.
Nonostante si disponga di una infrastruttura completamente funzionante gli sviluppi attuabili sono molteplici. L'adozione di meccanismi di sicurezza e critografia è sicuramente uno di questi: la natura prototipale del progetto ha posto il focus principalmente sugli elementi strutturali, ma in fase di sviluppo si è comunque tenuto conto della problematica lasciando ampio spazio di manovra per una futura implementazione di misure a tutela della privacy degli utenti. Un altro possibile sviluppo può essere la pianificazione di un testing approfondito su dispositivi fisici, al fine di avere un riscontro più dettagliato sulle prestazioni effettive: Arduino e Raspberry offrono soluzioni ideali per questi scenari, economiche ma comunque completabili con componenti di sensoristica ed in grado di consentire l'esecuzione di programmi Java (i nostri Server). Infine la diversificazione dei tipi di client attraverso lo sviluppo di librerie per dispositivi non-Android consentirebbe di apprezzare ancor di più l'estensibilità della piattaforma, permettendone l'utilizzo ad un numero maggiore di utenti.