



Università degli Studi di Udine
Facoltà di Ingegneria e Architettura
Corso di laurea di Ingegneria Elettronica

SISTEMA DI INTERFACCIAMENTO BASATO SU BOT PER APPLICAZIONI STRUTTURATE A MICROSERVIZI

Relatore: Pier Luca Montessoro

Laureando: Riccardo Fontanini

Anno Accademico 2016/2017

BOT

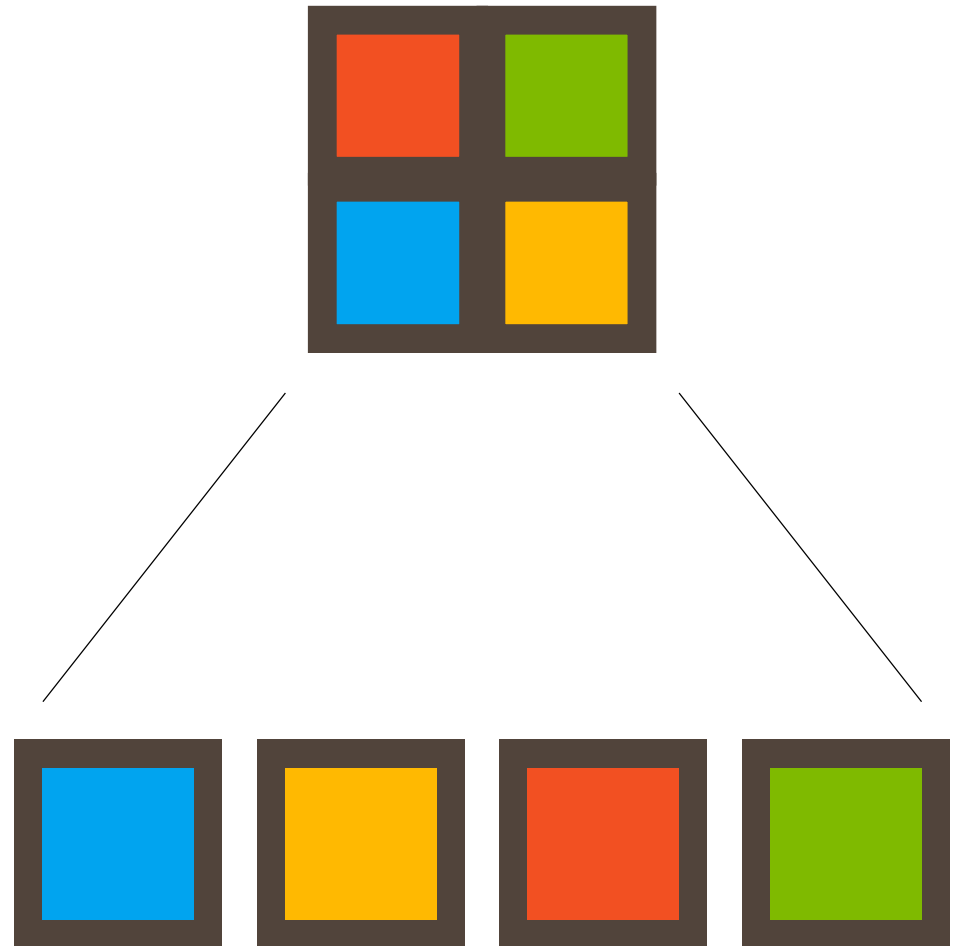
«Programma che accede alla rete attraverso lo stesso tipo di canali utilizzati dagli utenti umani ed interagisce con essi attraverso un linguaggio comprensibile fornendo servizi o attuando delle operazioni.»



Microservizi

«In short, the microservice architectural style is an approach to developing a single application as a suite of small services, each running in its own process and communicating with lightweight mechanisms, often an HTTP(s) resource API.»

Martin Fowler



Obiettivo della tesi

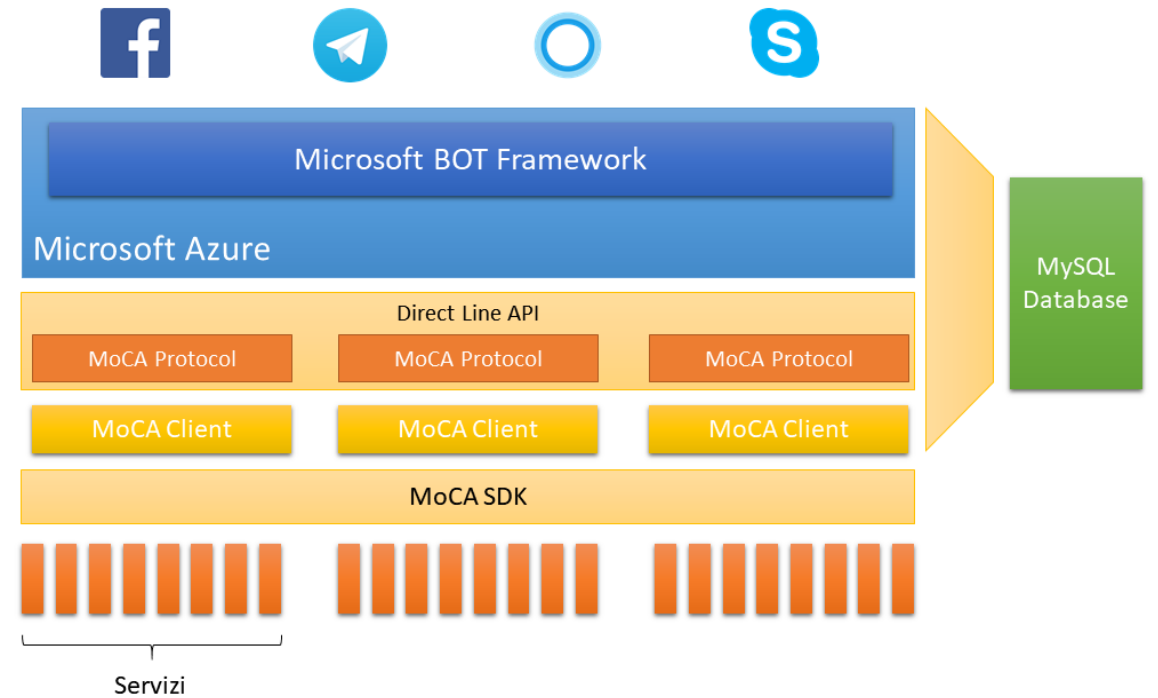
L'obiettivo della tesi è quello di progettare e realizzare un BOT che sfrutti i Social Network per eseguire e/o interagire con i programmi realizzati dagli studenti durante il corso di Fondamenti di Programmazione.

Caratteristiche del sistema

- Facile da usare e da capire dagli studenti
- Impatto minimo per lo studente all'interno del proprio codice
- Non includere la necessità di nozioni ulteriori a quelle fornite dal corso di studi
- Funzionare su ambienti che non necessitano di ulteriori caratteristiche rispetto a quelle utilizzate dagli studenti per programmare
- Scalabile
- Indipendente: manutenzione periodica non necessaria
- Multi linguaggio

Sistema generale

I servizi si interfacciano attraverso i MoCA Client verso il Microsoft BOT Framework, che a sua volta li collega con i vari Social Network. Come si vede i servizi sviluppati dagli studenti sfruttano il MoCA SDK per interfacciarsi al client MoCA.



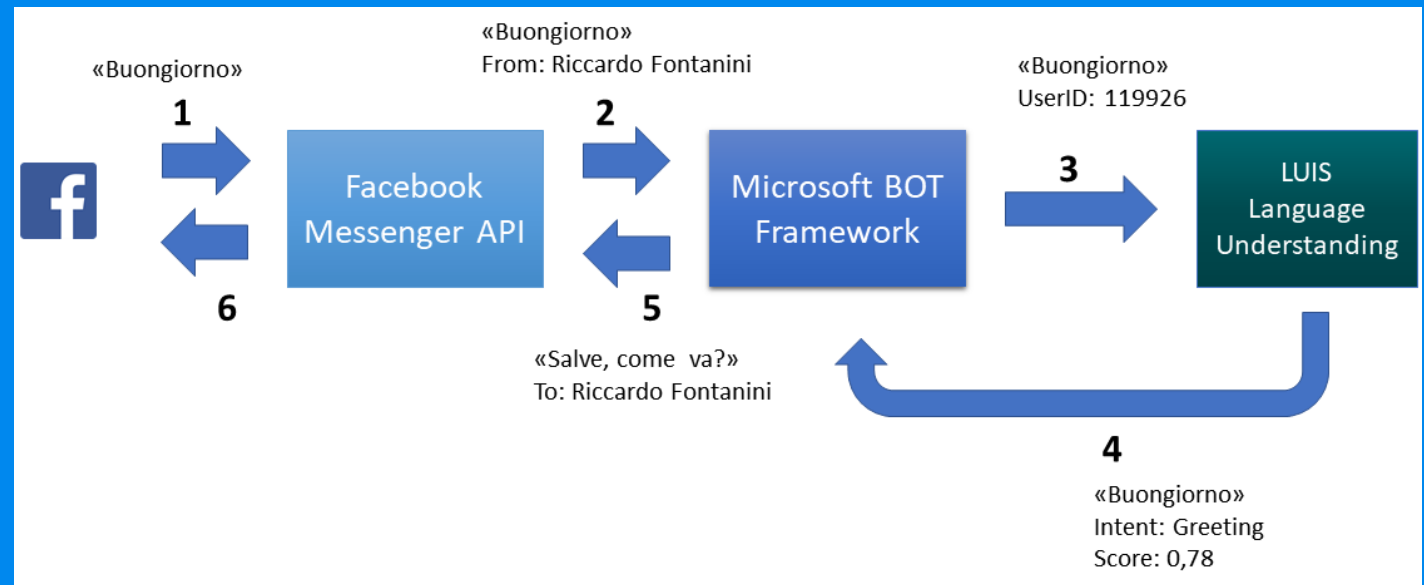
Tecnologie utilizzate

- BOT sviluppato in C# ed istanziato su Microsoft Azure
- Client MoCA sono stati realizzati in C
- SDK implementate in C ma estensibili a C++, C#, JAVA, Python, NODE.JS, PHP, RUBY
- Database MySQL + interfaccia in PHP
- Il protocollo utilizzato maggiormente per interconnettere i vari sistemi è HTTPs (crittografato con TLS 1.2)



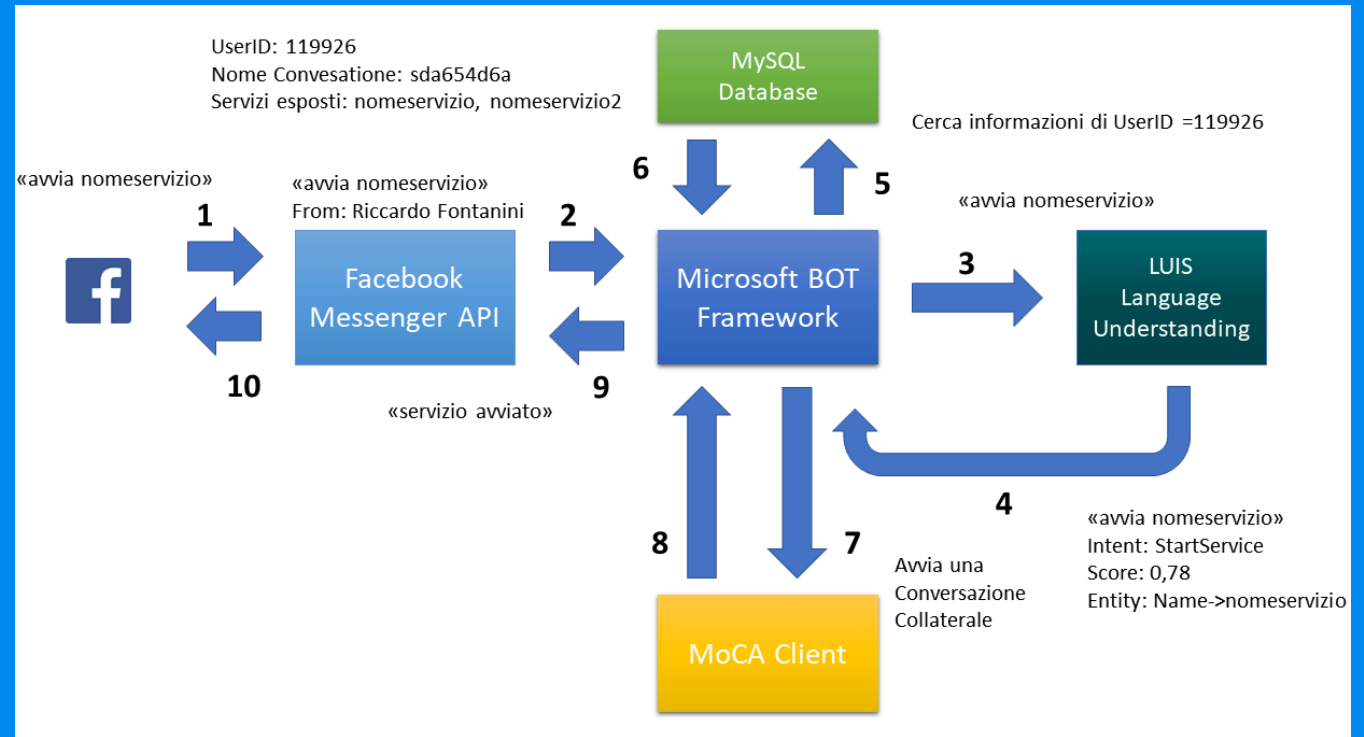
Dialogare con un BOT

Quando un messaggio viene ricevuto dal BOT, viene inoltrato al sistema LUIS (Language Understanding Intelligent Service) di Microsoft adibito al riconoscimento e all'interpretazione delle frasi inserite.



Trasmissione delle informazioni

Un semplice comando come l'avvio di un servizio per essere interpretato dal sistema deve passare attraverso un numero considerevole di sottosistemi e protocolli.



MoCA (Montessoro Communication Application)

```
riccardo@pc-Riccardo: /mnt/d/Repository/moca/build/linux

          _ _ _ _ _
         / / / / /
        / / / / /
       / / / / /
      / / / / /
     / / / / /
    / / / / /
   / / / / /
  / / / / /
 / / / / /
/ / / / /

This program is made by Riccardo Fontanini

UserID: 119926
Now I'm looking into your files to discover new programs...
Found: [ciao]
Found: [sendandreceive]
Total folders found: 2

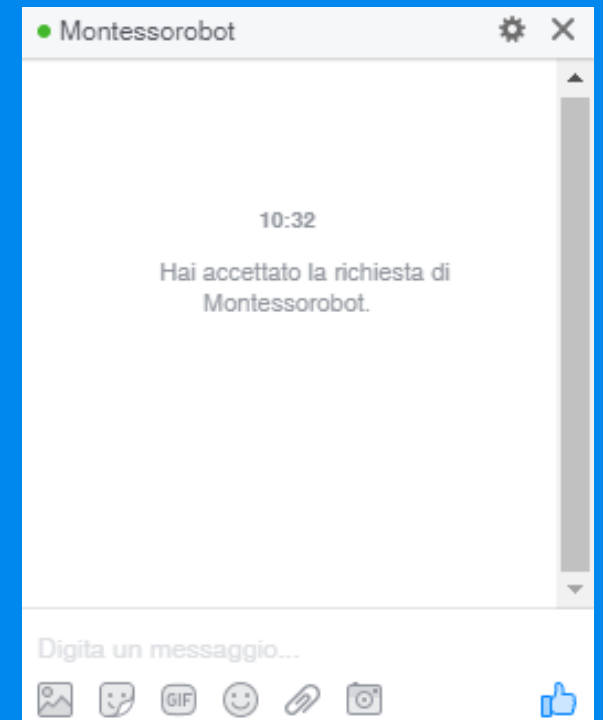
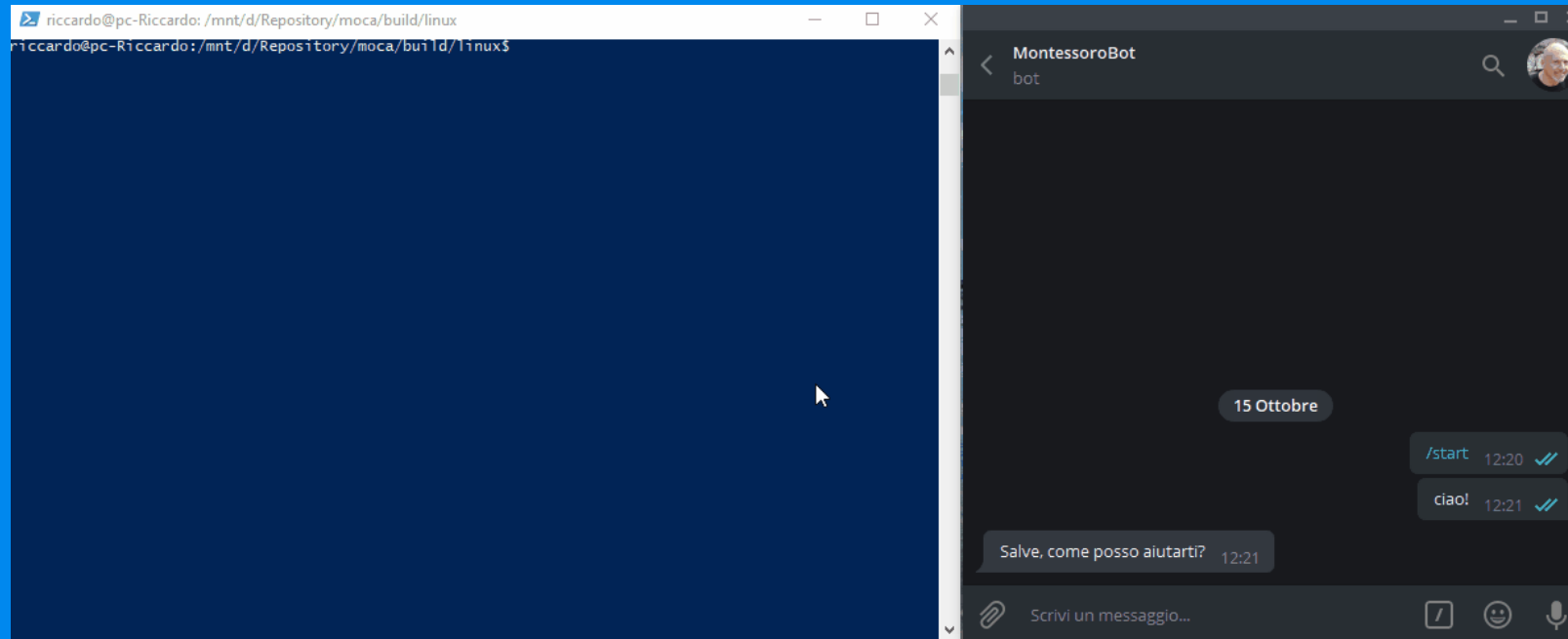
Service found: ciao/ciao
Service found: sendandreceive/sendandreceive
All folders are checked... And it's OK! Wonderful
Main process has PID: 17
Starting service: Allocating memory for services.. DONE.      Can use 50000 service at the same time!

Creating new conversation with MontessoroBOT.. DONE.
Sending information to bot.. DONE.
Creating Poller Service... DONE.
Creating collateral conversation poller... DONE.
Publishing informations to Database... DONE.

-----

Created collateral with Token: Ay_aHU8nWyk.dAA.TABMAHoAQgBqAG8ATgByAEoAccBzADEAMABCAGQAbgByADgAYQBvADgA
eAA.YQvTcrZC0wE.kPIomkaktYE.JpNiVzBeewV1XyRhqXS7jKSSplvqujEZLNSpE1GLvik
Found free service 0
Inserted name in services 0
Starting service 0...
Created pid 48
Created pid 0
Inside child process
file scritto
sendandreceive letti: 18 Messaggio di prova
Writing message: come va
```

Esempi applicativi



Ricapitolando

Sono stati rispettati i presupposti di

- Scalabilità
- Estensibilità
- Indipendenza
- Facilità d'uso
- **Ridondanza**
- **Sicurezza**
- **Facilità di impiego in campi diversi**

Possibili sviluppi

- Modello economico per lo sviluppo di applicazioni basate su questo progetto
- Integrazione con sistemi di distribuzione software esistenti (Docker)
- Implementazioni di algoritmi aggiuntivi e funzioni per rendere il BOT più semplice ed efficace
- Estensione delle SDK per gli studenti e i sistemi operativi utilizzabili

Confronto tra due programmi I/O

```
#include <stdio.h>
#include <stdlib.h>
#include "MoCAAPI.h"

int main(int argc, char **argv) {
    if(argc < 2)
        return -1;
    //GET first parameter and setup communication
    initMocaRW(argv[1]);
    //write message TO bot
    mocaWrite("Messaggio di prova", _MOCAWRITEINBOT);
    char buffer[500];
    //read message from bot
    mocaRead(buffer, 500, _MOCAREADFROMBOT);
    //retransmit message to bot
    mocaWrite(buffer, _MOCAWRITEINBOT);
    //close communication
    mocaClose();
    return 0;
}
```

```
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char **argv) {

    //write message TO bot
    printf("Messaggio di prova\n");
    char buffer[500];
    //read message from stdin
    fgets(buffer, 500, stdin);
    //retransmit message to bot
    printf(buffer);

    return 0;
}
```

Grazie per l'attenzione