

Connettere la propria Alexa Skill con ROS

Davide Ferrari

Università di Modena e Reggio Emilia

In questo documento verrà descritta la procedura per collegare una *custom skill* di Alexa con un nodo ROS.

Sarà necessario sviluppare un *custom server* che funga da *endpoint* per la nostra skill e permetta il collegamento tra il *front-end*, ovvero la parte di riconoscimento e analisi vocale, e il **back-end** della skill, ovvero la parte di codice nel quale i dati vengono processati e integrati in ROS. Il backend dovrà essere integrato in ROS e avrà il compito di gestire lo scambio di informazioni tra la skill e l'ambiente ROS.

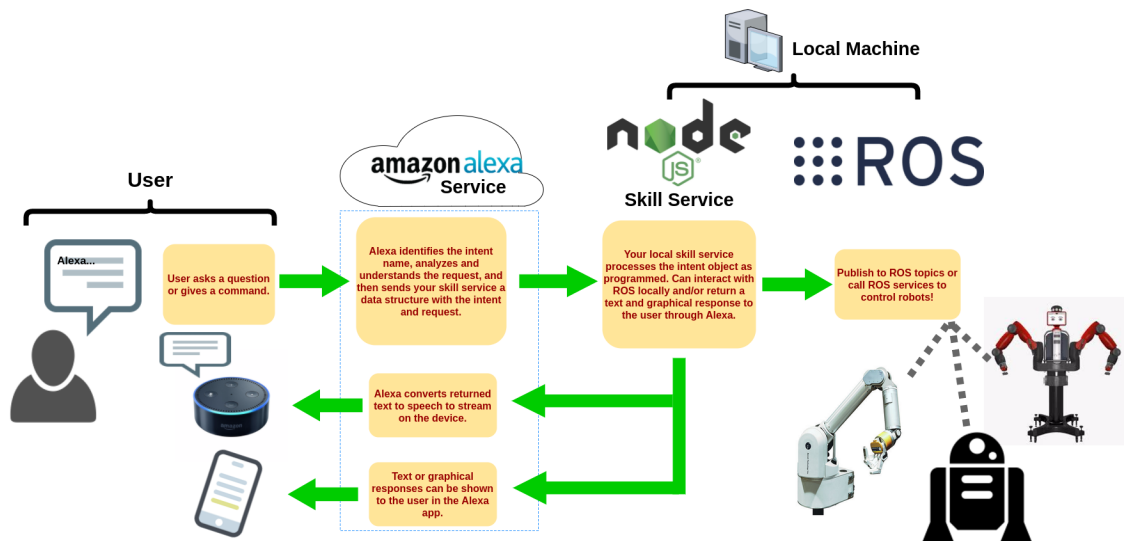


Figura 1: Architettura della Connessione Alexa - ROS

Indice

1	Creazione del Custom Server	1
2	Sviluppo della Alexa Skill	2
3	Creazione del Back-End in ROS	4
4	Guida di Utilizzo	6

1 Creazione del Custom Server

Affinché le richieste di Alexa raggiungano il server locale, la rete deve essere configurata per il tunneling del traffico HTTPS verso una porta specifica sul computer locale. Per fare ciò si può utilizzare **ngrok**, implementando la modalità tunnel.

Per prima cosa è necessario impostare la variabile d'ambiente **ROS_IP** uguale all'indirizzo IP del computer:

- Creare la variabile **IP** nel *.bashrc* e assegnarle il seguente valore:

```
IP="$ (ip a | grep "scope global" | grep -Po '(?<=inet )[\d.]+)'"
```

- Effettuare l'export di **ROS_IP** nel terminale prima di lanciare il server:

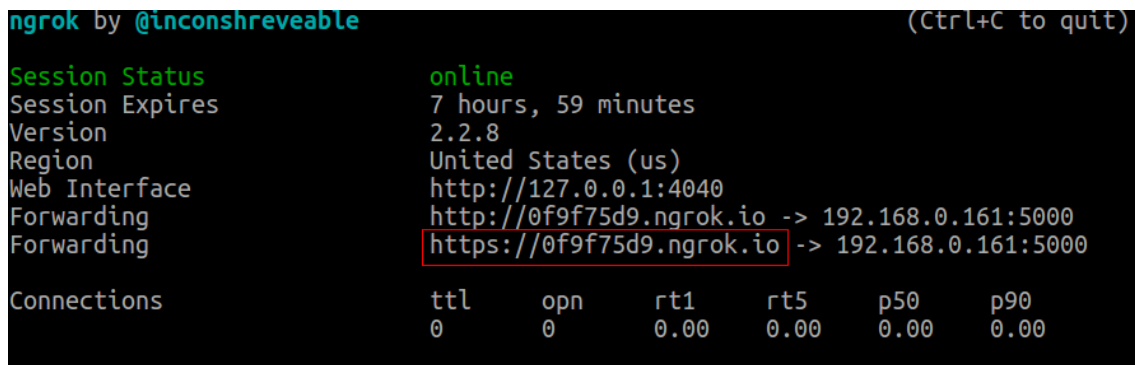
```
export ROS_IP=$IP
```

Ora si può procedere al download di **ngrok** per Linux e installarlo, inserendo il personal token ricevuto in seguito alla creazione di un account ngrok:

```
./ngrok authtoken personal_token
```

Per avviare il tunnel HTTPS lanciare il seguente comando:

```
./ngrok http $ROS_IP:5000
```



```
ngrok by @inconshreveable (Ctrl+C to quit)

Session Status      online
Session Expires     7 hours, 59 minutes
Version              2.2.8
Region               United States (us)
Web Interface        http://127.0.0.1:4040
Forwarding            http://0f9f75d9.ngrok.io -> 192.168.0.161:5000
Forwarding            https://0f9f75d9.ngrok.io -> 192.168.0.161:5000

Connections          ttl    opn    rt1    rt5    p50    p90
0                    0      0.00   0.00   0.00   0.00
```

Figura 2: Server ngrok. In evidenza l'indirizzo https del server.

2 Sviluppo della Alexa Skill

Accedi alla pagina Amazon Developer Console, ambiente di sviluppo delle skill per Alexa e seleziona "*Create Skill*". Inserisci il nome della skill e seleziona un modello personalizzato.

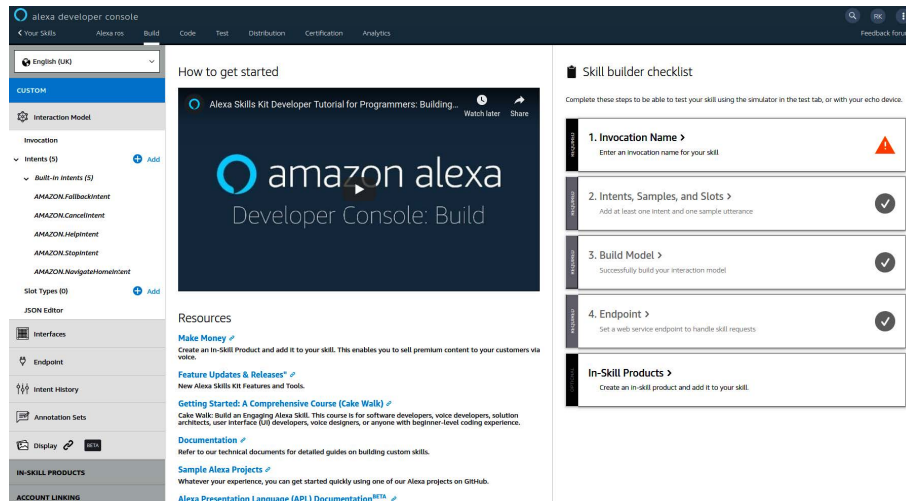


Figura 3: Alexa developer account page.

Configurare il nome di invocazione e gli intenti desiderati, con le opportune *utterances* e *slots* per salvare le variabili:

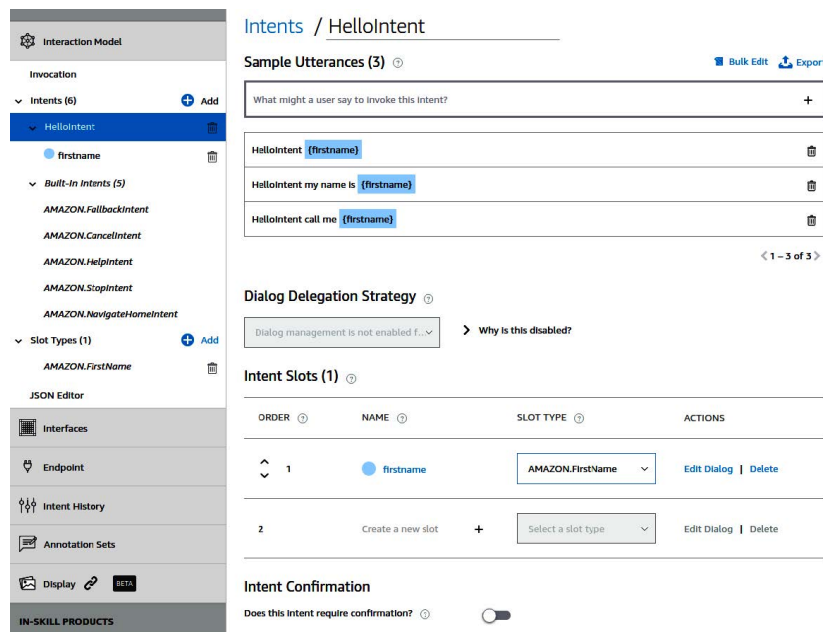


Figura 4: Intents.

Salvare il modello e buildare utilizzando il pulsante *Build Model*.

Una volta completata la compilazione, andare nella sezione *Endpoint* sul lato sinistro e seleziona il pulsante di opzione HTTPS. Immettere il link dell'endpoint https fornito da ngrok al lancio del server e, sotto la voce SSL Certificate, selezionare "*My development endpoint is a sub-domain of a domain that has a wildcard certificate from a certificate authority*".

3 Creazione del Back-End in ROS

Il *backend* è la parte di codice necessaria a elaborare gli input derivanti da ogni *intent* dell'analisi vocale e fornire le rispettive risposte sfruttando il tunnel creato da ngrok. Può essere scritto in diversi linguaggi (C++, Node.js, Python), in questo esempio si è scelto di sfruttare un'espansione di Python: *Flask-Ask*.

Flask-Ask è un nuovo micro-framework Python che abbassa notevolmente il livello per lo sviluppo delle skill di Alexa. Flask-Ask è un'estensione Flask che rende semplice e divertente la creazione di interfacce utente vocali con Alexa Skills Kit. Di seguito la repository ufficiale del progetto oppure la [Guida Ufficiale Alexa](#):

`https://github.com/johnwheeler/flask-ask`

Per iniziare, avrai bisogno di Python installato. Se utilizzi una versione recente di OS X o Linux, Python viene preinstallato. Potrebbe anche essere necessario installare pip, che può essere trovato [qui](#). Una volta installati Python e pip, apri un terminale e digita il comando seguente per installare Flask-Ask:

```
pip install flask-ask
```

Una applicazione base Flask-Ask è composta nel seguente modo:

```
from flask import Flask
from flask_ask import Ask, statement

app = Flask(__name__)
ask = Ask(app, '/')

@ask.intent('HelloIntent')
def hello(firstname):
    speech_text = "Hello %s" % firstname
    return statement(speech_text).simple_card('Hello', speech_text)

if __name__ == '__main__':
    app.run()
```

- L'oggetto *Ask* viene creato passando l'applicazione Flask e una strada a cui inoltrare le richieste Alexa.
- La dichiarazione dell'intento associa *HelloIntent* a una funzione di visualizzazione *Hello*.
- Lo slot *firstname* dell'intento è implicitamente associato al parametro *firstname* di *Hello*.
- Infine, un builder costruisce una risposta vocale e visualizza una scheda contestuale nell'app per smartphone / tablet Alexa.

4 Guida di Utilizzo

- Avviare il tunnel ngrok:

```
./ngrok http $ROS_IP:5000
```

- Copiare il link del tunnel nell'endpoint della skill Alexa e fare il Build del Modello (se si utilizza un account ngrok free, è necessario ripetere l'operazione ogni volta che viene riavviato il server).

- Avviare il server (backend):

```
roslaunch flask_ask_ros start_skill_server.launch ngrok:=true
```

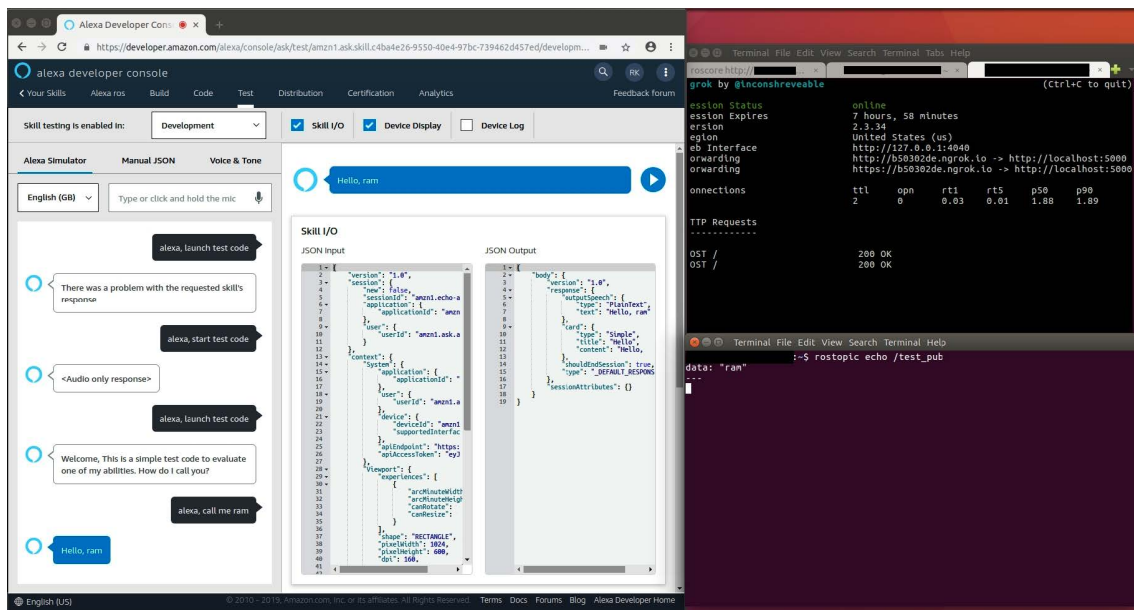


Figura 5: Test della Skill.