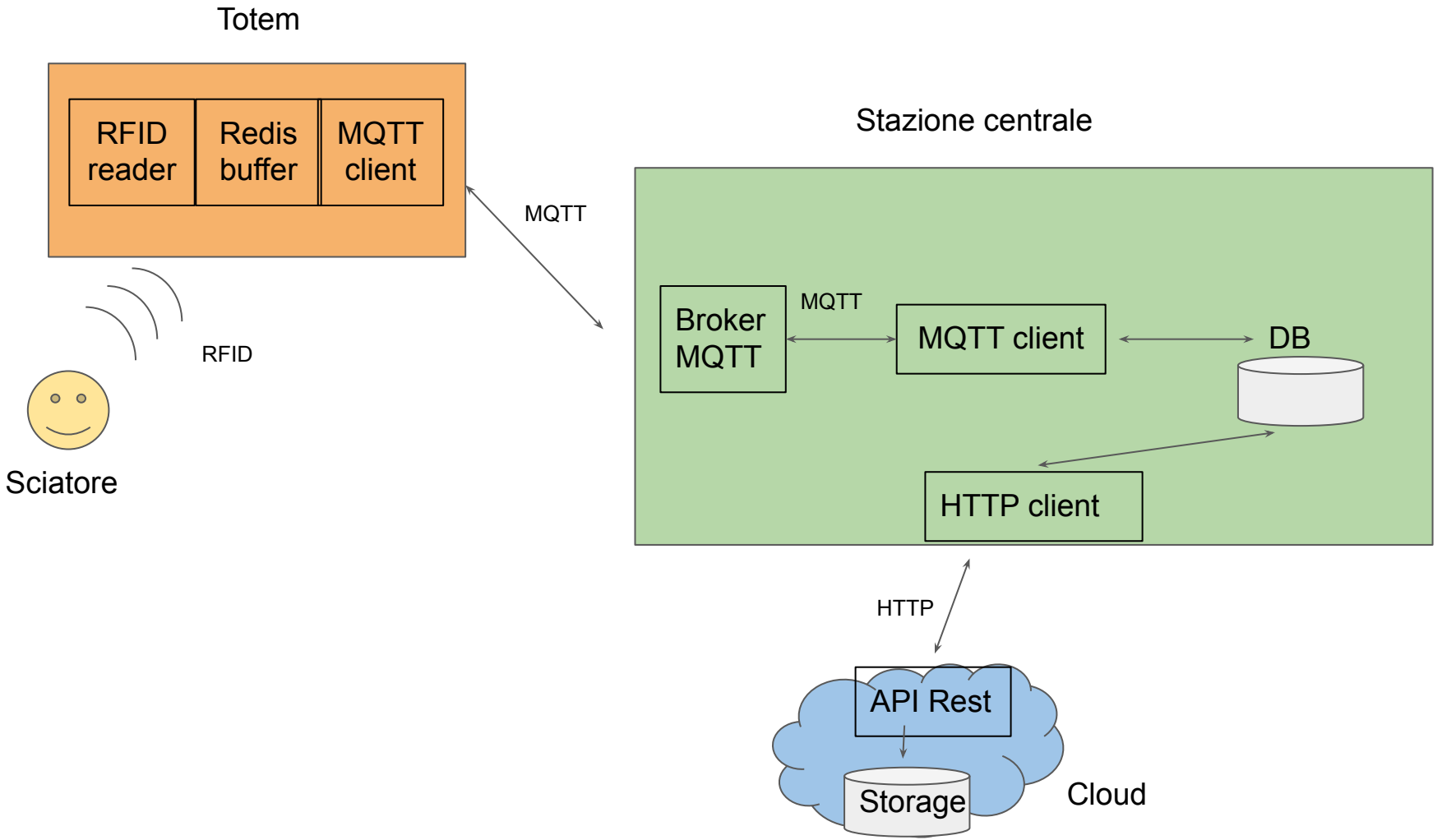
Totem
RFID reader
Redis buffer
MQTT client
MQTT
RFID
Sciatore
Stazione centrale
Broker MQTT
MQTT
MQTT client
DB
HTTP client
HTTP
API Rest
Storage
Cloud

Totem

Non ha problemi di batteria quindi non ho necessità di utilizzare un protocollo estremamente leggero come coap. Inoltre per coap rappresenterebbe un problema la gestione degli IP dinamici. Per questo scopo andrebbe bene sfruttare MQTT poiché il totem sarebbe un client che pubblica messaggi ad un broker. Se il totem perde connessione salva i passaggi non inviati su di un buffer di memoria che potrebbe essere implementato usando semplicemente redis (non serve gestione avanzata di code di AMQP poichè invio tutto a broker).

Il totem si connette al broker autenticandosi con username e password e permesso per publish e subscribe (totem potrebbe avere bisogno di ricevere messaggi dal broker, ad esempio per la sua attivazione/disattivazione). Inoltre fa connessione con Last Will Testament e pubblica un messaggio di last will al topic skiTotems/lastwill/totemID con messaggio 'TotemID disconnected'. Il connect viene fatto con clean session a false (voglio ricevere comandi inviati quando totem è disconnesso).

Il totem pubblica per ogni passaggio di sciatore un messaggio al topic skiTotems/passages/totemID con payload un json nella forma:

```
{
        SkierID: "RFID",
        Timestamp: "tempo di passaggio"
}
```

La QOS del messaggio va settata a 2 (devo assicurarmi che il passaggio venga registrato esattamente una volta).
Il totem fa subscribe al topic skiTotems/commands/totemID con QOS 2 per ricevere i comandi inviati da stazione centrale.

Stazione centrale

Il broker MQTT girerà nella stazione centrale. Sempre nella stazione centrale girerà un client MQTT. Questo client si connette con clean session a false per ricevere anche messaggi pubblicati sul broker mentre l'istanza del client non funzionava. Il client fa subscribe con wildcard a tutti i topic su cui i totem pubblicano i messaggi relativi ai passaggi (skiTotems/passages/#) e tutti i topic relativi ai last will testament (skiTotems/lastwill/#) per sapere se qualche totem ha dei problemi. Il subscribe a passages verrà fatto con QOS 2 per avere la stessa QOS con cui i passaggi vengono pubblicati dal totem.

Inoltre il client può pubblicare messaggi ai topic skiTotems/commands/totemID per inviare comandi ai totem con QOS 2 e payload una stringa con il comando da inviare.

I dati ricevuti con i messaggi ricevuti al topic skiTotems/passages/# vengono salvati sul db. Il database sarà un database time series adatto a ricevere le numerose scritture di dati con timestamp. Il timestamp sarà l'id mentre l'id dello sciatore e del totem saranno due tag per velocizzare i filtraggi in base a questi due id.

Prima di salvare i dati sul database la stazione centrale farà dei controlli per eliminare eventuali dati anomali (es. passaggi ravvicinati di uno sciatore ad uno stesso o più totem).

I dati importanti che richiedono di essere salvati a lungo termine verranno inviati ad uno storage in cloud tramite chiamata http ad una API. Si sceglie questo protocollo perchè la stazione centrale ha la potenza di calcolo necessaria per usarlo e per la sua semplicità di implementazione.