Gedistribueerde Systemen
iHome

Tim Leys,
Lisette van Leeuwen

22 december 2016

In dit verslag wordt kort besproken welke architectuur, algoritmes en libraries wij hebben gebruikt. Ook wordt beschreven hoe het programma gebruikt moet worden.
Devices die zelf een controller kunnen starten, zoals een user of een fridge, worden in dit verslag smart devices genoemd.

1 Communicatie

Tussen een client en een server zijn twee types communicatie mogelijk, namelijk synchroon en asynchroon. Bij synchroon moet de communicatie eerst voltooid zijn, voordat de code verder kan gaan. Bij asynchrone communicatie is dit niet het geval. Als de ontvanger niet actief is, zal bij asynchrone communicatie de zender wachten totdat de ontvanger terug actief is.
Wij hebben voornamelijk synchrone communicatie gebruikt. Enkel bij leader election gebruiken we asynchrone communicatie. Bij leader election wordt steeds het hoogste ID naar de volgende client in de ring verstuurt. Indien we synchrone communicatie zouden gebruiken, kan de functie die de election start pas verder gaan als een leader is verkozen. Met asynchrone communicatie zetten we meerdere threats op, waardoor de startende functie verder kan zodra het een bericht heeft verstuurd naar de volgende client in de ring.

2 Replicatie

Bij replicatie zijn we ervan uitgegaan dat AVRO inkomende berichten sequentieel en FIFO gewijs verwerkt.
Wanneer de data van de server wijzigt en de smart devices moeten hiervan op de hoogte gebracht worden, wordt deze verandering direct verstuurd naar de smart devices. Alle databanken van de smart devices worden geüpdatet voor de functie returned en de remote call eindigt.
Enkel de werkende server zal zijn data versturen naar alle geconnecteerde smart devices.
We kunnen hier spreken van sequentiële consistentie. Alle operaties worden in dezelfde volgorde verwerkt door de server als waarin ze aankomen. Doordat de controllers in de smart devices worden geüpdatet vooraleer er volgende message verwerkt worden, geldt dat die controllers in de juiste volgorde worden geüpdatet mits een kleine delay.

3 Fault tolerance

We willen dat de server ten allen tijde weet welk device nog online is, zodat er geen verbindingen worden gemaakt met gecrashte devices. Om dit te realiseren hebben we de failure detection methode heartbeat geïmplementeerd. Alle devices sturen drie keer per seconde een bericht naar de server, zodat de server weet dat het device nog online is.
Indien de server crasht, willen we dat een fridge of user de rol van controller overneemt. De user of fridge met de hoogste ID moet verkozen worden. Om te bepalen welke actieve client de hoogste ID heeft, wordt het ring algoritme van Chang-Roberts gebruikt. Zoals hiervoor beschreven, gebruiken we hier asynchrone communicatie voor.
Tegengesteld aan wat er in de opgave staat, kan een fridge of user dat uitverkozen is tot leader nog steeds gebruikt worden als fridge of user. De controller draait dan op de achtergrond.
Als de originele server gecrasht is, maar later terug online komt, willen we dat de originele server de rol van controller terug overneemt. Om te bepalen of de originele server weer online is, stuurt de leader drie keer per seconde een bericht richting de server volgens de ping methode. Indien het bericht aankomt, weten we dat de server weer online is. Direct wordt dan de data van de leader naar de server gestuurt en de server stuurt een bericht naar alle devices dat het terug online is. De communicatie verloopt dan gelijk aan wanneer de server niet was uitgevallen.

4 Code uitvoeren

Naast de benodigde libraries voor AVRO hebben wij ook een JSON library gebruikt.
In de bijgeleverde zip zitten vijf jar files. Dit zijn runnable jars en kunnen uitgevoerd worden met het commando `java -jar filename.jar`.