

A dark blue vertical bar runs down the left side of the page. A blue arrow points to the right from the bar, containing the date.

30-10-2021

# Opdracht 3

Event based concurrency met AKKA.

Several thin, curved lines in dark blue and light grey originate from the bottom left corner and sweep upwards and to the right.

Emanuel de Jong & Peter van Assenbergh

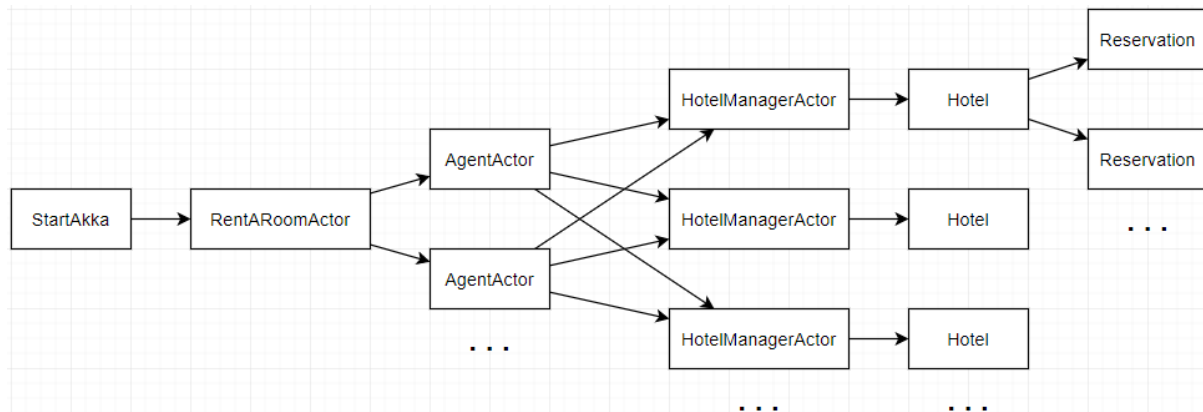
## Inhoud

Rent A Room .....	2
Systeem ontwerp .....	2
Technische details .....	3
StartAkka .....	3
Berichten .....	4
RentARoomActor .....	4
De receptionist .....	4
Info voor de RentARoomActor GroupRouter .....	4
Een HotelManagerActor lijst voor AgentActors .....	4
Fouten hantering .....	5
Input validatie .....	5
Fout weergave .....	5
Unit tests .....	5
Samenvatting .....	6

## Rent A Room

RentARoom.com heeft gevraagd een simulatie voor hun toekomstige website te maken om bekend te worden met het AKKA systeem. Dit document zal informatie geven over de simulatie, met name hoe het werkt.

## Systeem ontwerp



*De relatie tussen de actors en objecten*

In **StartAkka** word het systeem opgestart en word de input van de gebruiker doorgestuurd. Na het opstarten kan het als klant gezien worden. Deze klant kan een gast zijn die hotels wil bekijken en reservaties wil aanmaken en beheren, of een hotel vertegenwoordiger zijn die zijn hotel in het systeem wil zetten of verwijderen.

**RentARoomActor** Maakt AgentActors en stuurt de berichten van de klant naar ze door. RentARoomActor kan als het bedrijf worden gezien.

**AgentActors** werken als de medewerkers van Rent A Room. Ze kunnen hotels aanmaken en beheren en zoeken uit waar reservering berichten heen gestuurd moeten worden. Ook kunnen ze informatie weergeven. Voor deze taken hebben ze een lijst van HotelManagerActors en daarmee ook toegang tot alle hotels en reserveringen.

**HotelManagerActors** zijn als de medewerkers van de hotels zelf. Ze hebben ieder hun eigen hotel informatie. Met name de reserveringen.

Voor de duidelijkheid het volgende voorbeeld:

Jan wil bij Hotel Atlanta een reserveringsverzoek doen voor 2 kamers. Hij geeft dit via **StartAkka** aan de **RentARoomActor** door. De RentARoomActor zoekt een **AgentActor** uit die al een tijdje geen opdracht meer heeft gekregen en stuurt het verzoek er naar door. De AgentActor zoekt de **HotelManagerActor** op, die het hotel heeft dat genoemd word in het verzoek. Daarna stuurt het het verzoek naar de HotelManagerActor op. Als laatste zet de HotelManagerActor het verzoek in de lijst van reserveringen in het hotel object.

Bij dit voorbeeld zijn checks zoals of het hotel in het systeem zit en of de reservering mogelijk is weggelaten.

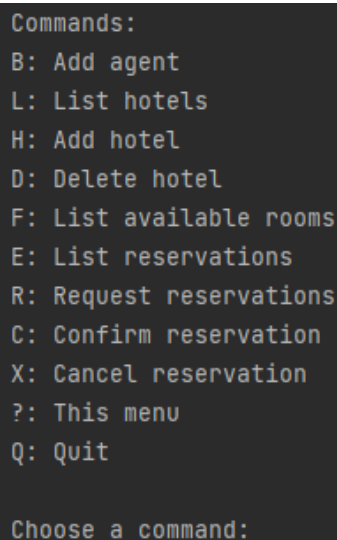
## Technische details

Hierbij nemen wij aan dat u al basis kennis over AKKA heeft. Is dit niet het geval, dan raden wij aan de volgende korte introductie te volgen: <https://developer.lightbend.com/docs/akka-platform-guide/index.html>

### StartAkka

Na het opstarten van het AKKA systeem, is StartAkka is een console applicatie die eindeloos de volgende loop volgt:

1. Vraag de gebruiker een commando te kiezen

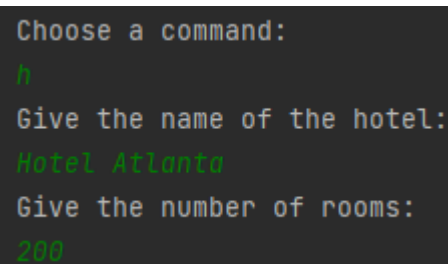


```
Commands:
B: Add agent
L: List hotels
H: Add hotel
D: Delete hotel
F: List available rooms
E: List reservations
R: Request reservations
C: Confirm reservation
X: Cancel reservation
?: This menu
Q: Quit

Choose a command:
```

*Screenshot van de te kiezen commando's*

2. Vraag eventueel naar data.



```
Choose a command:
h
Give the name of the hotel:
Hotel Atlanta
Give the number of rooms:
200
```

*Screenshot van ingegeven data*

3. Stuur een bericht (zie XXX) naar de RentARoomActor.
4. Wacht op een antwoord (in tekst formaat).
5. Laat dit antwoord aan de gebruiker zien.

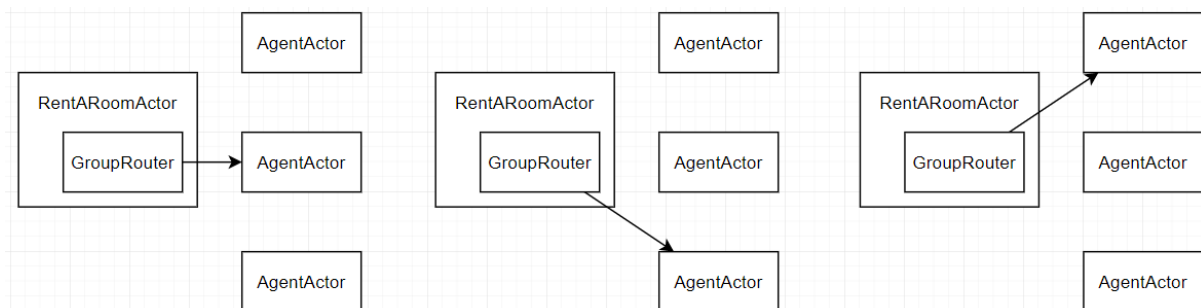
StartAkka is het gedeelte dat het meest aangepast zal moeten worden als de simulatie in een web applicatie getransformeerd zal worden. Het is namelijk het startup proces, de API en de front-end in een en geeft toegang tot functionaliteiten voor gasten, medewerkers, systeem beheerders en hotel vertegenwoordigers zonder onderscheid tussen rechten te maken.

## Berichten

Zoals gebruikelijk in AKKA, word informatie tussen de actors gedeeld met berichten. In dit systeem heeft elk commando een uniek bericht. StartAkka krijgt altijd als antwoord een bericht van het type Response waar tekst voor de gebruiker in zit. Verder is er op het moment alleen een RequestHotel bericht van een AgentActor naar een HotelManagerActor en een SendHotel bericht terug naar de AgentActor zodat AgentActors hotels kunnen opvragen.

## RentARoomActor

De RentARoomActor word alleen gebruikt om nieuwe AgentActors aan te maken. Het heeft een GroupRouter waarmee AgentActors in runtime toegevoegd kunnen worden. Dit is handig voor schaalbaarheid. De GroupRouter gebruikt round robin routing om berichten naar de AgentActors te sturen. Dit betekend dat de AgentActors in een rij staan en elk nieuw bericht telkens naar de volgende actor gestuurd word:



*De volgorde waarop de GroupRouter berichten naar AgentActors stuurt.*

## De receptionist

De AKKA receptionist word op twee plekken gebruikt.

### Info voor de RentARoomActor GroupRouter

De GroupRouter moet weten als er een AgentActor toegevoegd of verwijderd word. Dit word gedaan met een statische ServiceKey in het AgentActor object. De GroupRouter is hierop gesubscribed en AgentActors registreren zich ervoor via de receptionist.

### Een HotelManagerActor lijst voor AgentActors

AgentActors hebben ieder een dynamische lijst van HotelManagerActors. Hoe dit word gedaan is vergelijkbaar met de tekst hierboven. Echter word hier de lijst zelf bijgehouden. AgentActors subscriben op de ServiceKey van het HotelManagerActor object en krijgen hiermee berichten als hotel managers gemaakt en verwijderd worden.

## Fouten hantering

Er is veel tijd besteed aan het vermijden en vinden van fouten in het systeem. Dit zorgt ervoor dat het systeem robuust blijft en nieuwe fouten snel opgelost kunnen worden.

### Input validatie

Alle gebruiker input word op basis fouten getest. Zo word er bijvoorbeeld gekeken of nummers echt nummers zijn, kunnen datums alleen in de toekomst zijn en kan er een minimum en maximum lengte van een tekst gezet worden.

Als de input niet gelid is, word er verteld wat er mis is en kan er opnieuw input worden gegeven. Dit herhaalt totdat de input wel geldig is.

### Fout weergave

Als er na het sturen van een bericht iets fout is, word dit aan de gebruiker verteld. Denk hier bijvoorbeeld aan als er geen hotel met de ingegeven naam bestaat of als er niet genoeg kamers zijn voor de reservering.

### Unit tests

Elk commando heeft ten minste één good weather test en veel van de bad weather gevallen zijn ook voorzien van een test. Deze testen kunnen automatisch aangezet worden om zonder moeite de applicatie op elk moment te kunnen testen. Hiermee kunnen fouten en ongewenste aanpassingen snel gevonden worden.

LET OP: Alle testen hebben een initialisatie periode van 1 seconde. Dit is om een bug te omzeilen waarbij berichten naar de RentARoomActor worden gestuurd voordat de eerste AgentActor zich bij de receptionist heeft gemeld.

## Samenvatting

Met behulp van AKKA werkt de simulatie voor het systeem van Rent A Room snel en schaalbaar. Alle fundamentele principes van AKKA zijn op een realistische manier gebruikt en er is genoeg kennis opgedaan om aan het hoofd product te kunnen beginnen.