**Development Exercise**

To solve this exercise I have decided to use Akka and Scala. For build and project maintaining I'm using SBT, test are implemented with Mockito and Akka TestKit.  
  
  
I have divided project into following modules:  
  
1) taxi-system – module which is handling location call messages and routing them to different parts of the system.  
  
2) management – center – module responsible for communication with external management system.  
  
3) gps-system – module for handling location requests.

The reason why I decided to split my code into modules is to improve flexibility and allow to deploy each module separately. Each module is build to separate jar file and contains main class. According to specification system is running in single JVM, but in the future we can easily split it by changing only Akka configuration file and running main class included in each module.

All actor dependencies are resolved by names, which are taken from configuration file. This will help us use remote actors without code refactoring.

Unclear part of the system for me was Management-Center I was not sure if this is a part of Taxi System or it is remote. That's why I have decided to use Adapter Design Pattern, this will help us to hide connection details, and if Management-Center module is remote we have to change only one class and we do not need to break API.

Assuming that Management-Center and TubeStationLocator are remote modules, calls to those systems can be time consuming and can slow down whole system, that is why all external requests are wrapped with Future object which allow to run multiple asynchronous calls.

All classes are testes with Unit Tests, external dependencies are mocked with Mockito or stubbed with simple implementation.  
  
  
Command to run program:  
  
*sbt clean run*  
  
Command to execute tests:  
  
*sbt clean test*  
  
Best regards,

Marcin Czech.