Distributed Systems (H0N08A)

Report: –

Tejas Narayana *(r0775819)*
Yoshi Vermeire *(r1234567)*

October 30, 2020

Question 1 The renter reaches out to the agency to check which types of cars are available during a certain period. Then the agency queries the registered companies to check availability. This process is illustrated in figure 1 and Figure 2

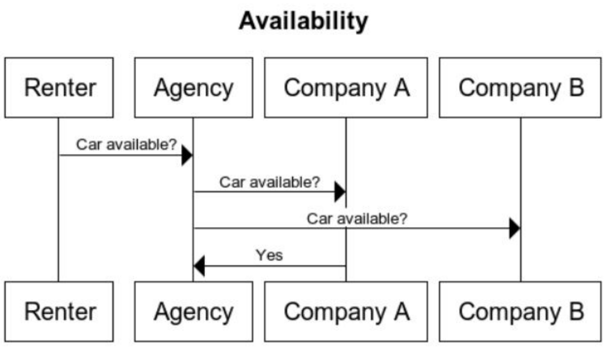

Figure 1: Check availability

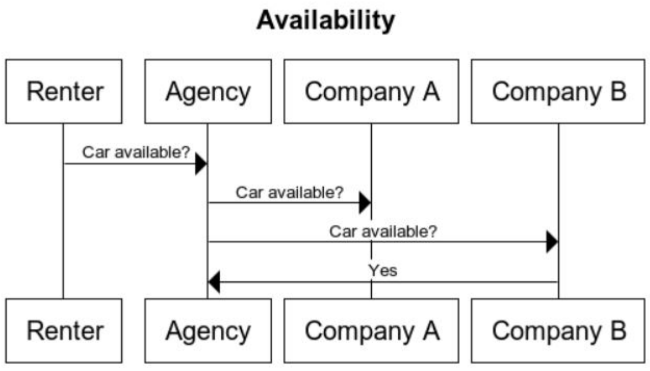

Figure 2: Check availability

Then, the agency correctly answer the renter. Based on the received information, the renter requests the agency to create a quote, as in figure 3 and Figure 4.

The first two steps can be repeated a few time, until the renter has created all the desired quotes. Thereafter, the renter tries to confirm the quotes to the agency. The agency then recursively tries to confirm all the quotes at the different companies. In a successful case all the quotes are confirmed as in Figure 5.

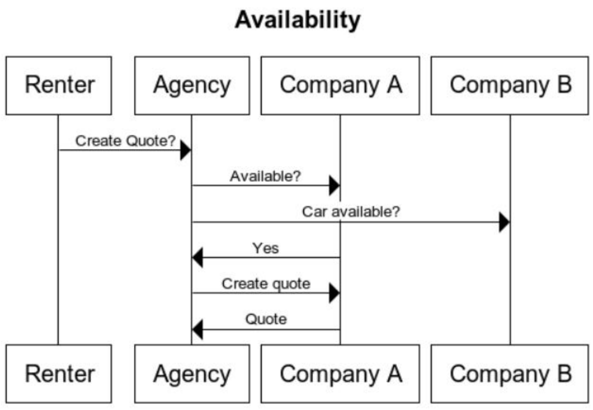

Figure 3: Create quote

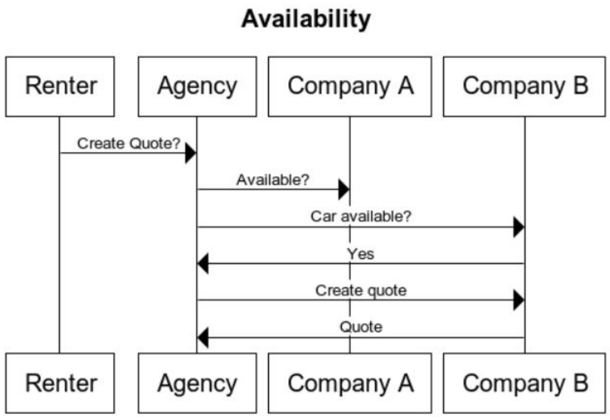

Figure 4: Create quote

However if the not all quotes can be successfully confirmed, the agency rolls back all the already confirmed quotes as in Figure 6.

Question 2 An object should be serializable if the complete state of the object is relevant to more than one party of the distributed system, and this state needs to be exchanged between the interested parties. For example, both renters and the car rental companies are interested in all the data of a quote.

Question 3 A class should be remotely accessible if multiple parties of a distributed system need to modify or read the state of the object, but it is infeasible to transmit the complete object, such as a CarRentalCompany.

Question 4 If a manager of the agency request the number of reservation of a certain renter, the manager (client) sends the operation that the manager wants the server to execute, the object on which the operation should be executed, and the name of the renter as the argument. Then, the server responses with the correct number.

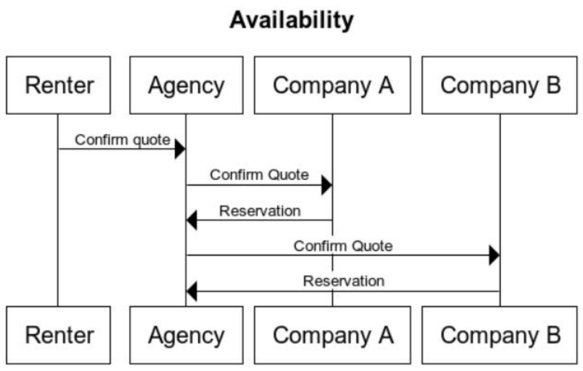

Figure 5: Successful reservation

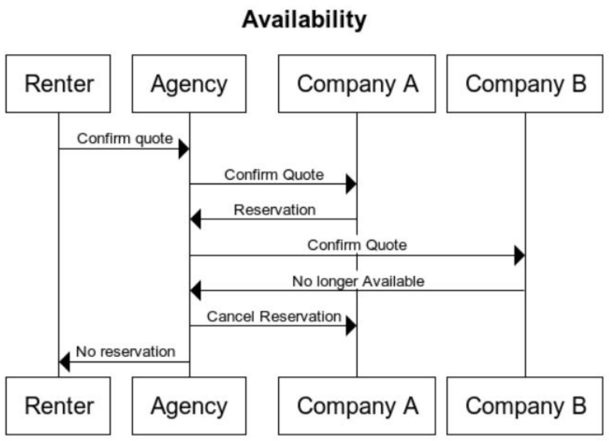

Figure 6: Failed reservation

Question 5 The client object is running on the client side, a CarRentalAgency object is running on the servers of the agency, and a CarRentalCompany is running on the servers of the rental company. The objects that represent the different cars of the company also run on the servers of the rental company, since the rental company is the only entity that should be able to read or alter the state of a car. The sessions run on the servers of the agency.SO, the agency can store the quotes for the client in case the client does not confirm the quotes right away.

Question 6 The naming service stores the names of all the registered companies. If the agency needs the actual object, the naming service looks up the reference in the RMI registry. We implemented the storage of the names of the companies separately from the registry, since it is possible that the agency has a contract with a company that is not online, and is hence not visible in the registry.

Question 7 A session is created via the agency, and the state of a session is stored on the server of the agency. At the end of the session, the client closes the session. If a session is closed, the reference to the session is removed from RMI registry. We store the sessions at the side of the agency, makes it easier to enforce that all operations happen through this section.

Question 8 Java RMI is not thread safe by default, because it is difficult for the framework to predict which objects should be blocked to avoid race conditions. It is more efficient to let the programmer who has specific knowledge about the application, handle the concurrency control.

Question 9