Saxion University of Applied Sciences

Bachelor Software Engineering
Parallel Computing

Distributed computing
RabbitMQ – Assignment 3

Student 548535 Mamedov Ignat

Deventer
2024

System Design

The system is architected as a distributed application composed of three primary types of processes:

1. *Clients*: Users interacting with the system to perform booking operations.
2. *Rental Agents*: Intermediary processes that handle client requests, manage reservations, and communicate with buildings.
3. *Buildings:* Representations of physical buildings that manage the availability of their conference rooms.

Components and Their Roles

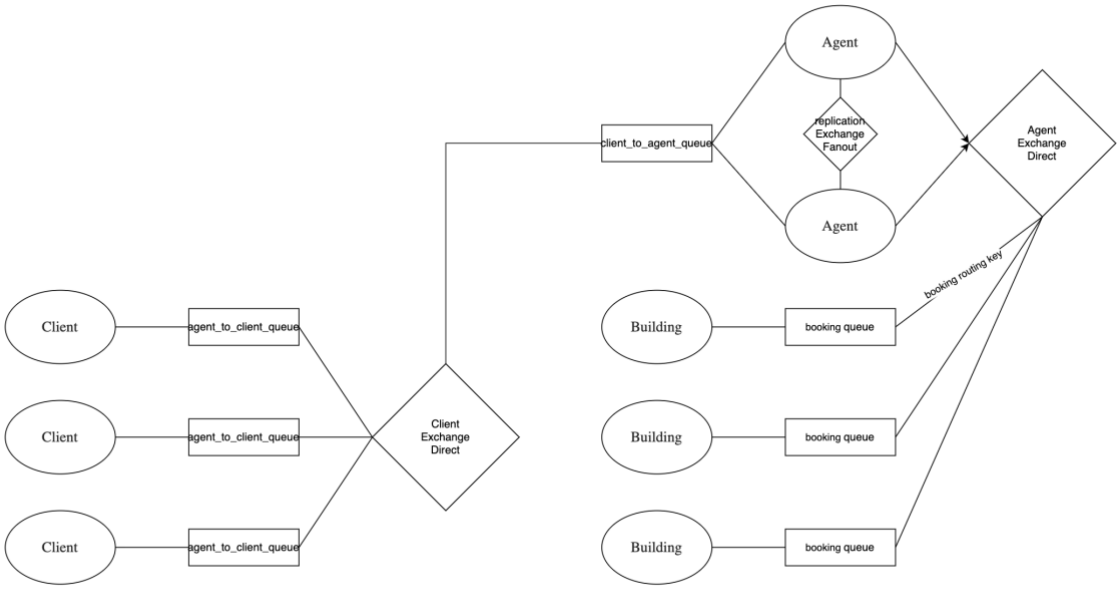
- Clients:
  - Initiate requests for building lists, bookings, confirmations, and cancellations.
  - Do not directly communicate with buildings; interaction is mediated by rental agents.
  - Are unaware of the specific rental agent they are communicating with.
- Rental Agents:
  - Receive and process client requests.
  - Maintain records of reservations and unconfirmed bookings.
  - Communicate with buildings to update room availability.
  - Synchronize booking data with other agents to ensure consistency across the system.
- Buildings:
  - Manage the availability of conference rooms.
  - Process booking and cancellation requests from rental agents.
  - Can be added to the system dynamically at runtime.

Communication Mechanism :

- Clients to Rental Agents: Clients send requests to rental agents via a direct exchange, using a common routing key.
- Rental Agents to Clients: Agents send responses back to clients using the client's unique identifier as the routing key.
- Rental Agents to Buildings: Agents send booking and cancellation requests to specific buildings using the building's name as the routing key.
- Buildings to Rental Agents: Buildings broadcast status updates to all agents using a fanout exchange.
- Agent Replication: Agents synchronize booking data among themselves using a replication exchange, ensuring data consistency.

More details on how the system elements interact with each other are shown in the diagrams below.Diagrams Description

Figure 1. Room booking



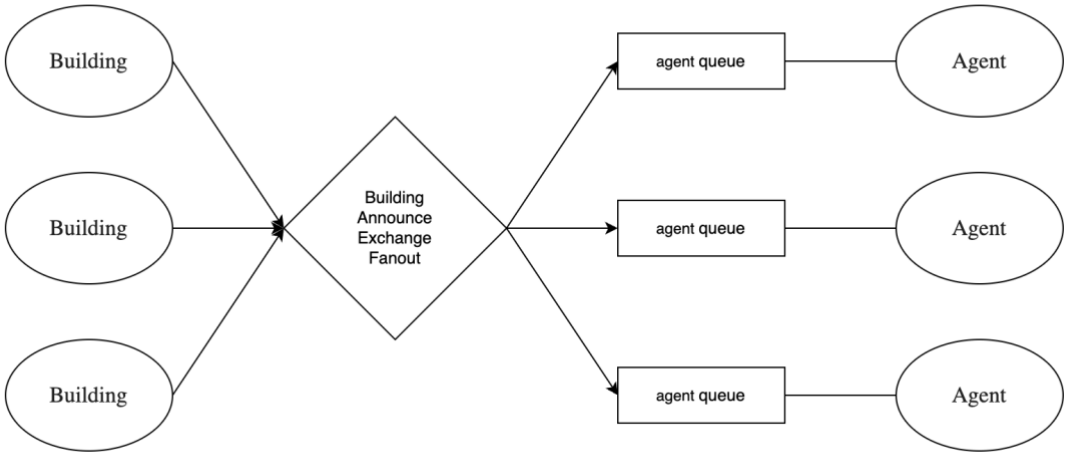
Main logic of System Communication:

- *Clients to Agents*:
  - Clients publish messages to client_exchange with routing key "client_to_agent".
  - Messages are routed to client_to_agent_queue, which agents consume from.
- *Agents to Clients*:
  - Agents publish responses to client_exchange with routing key equal to the client's clientId.
  - Messages are routed to agent_to_client_queue_{clientId} for the specific client.
- *Agent to Building:*
  - Booking/Cancellation: Agent publishes to direct_exchange with buildingName.
  - Building consumes from building_queue_{buildingName}.
- *Agent Replication:*
  - Agents publish booking data to replication_exchange.
  - Messages are broadcast to replication_queue_{UUID} for each agent, ensuring data synchronization.

It is also worth noting that information about buildings is stored on Agents with a timestamp. If no updates are received for an extended period, the

building is removed from the list of available ones, and a status update request is sent to it.

Figure 2. Adding new Building

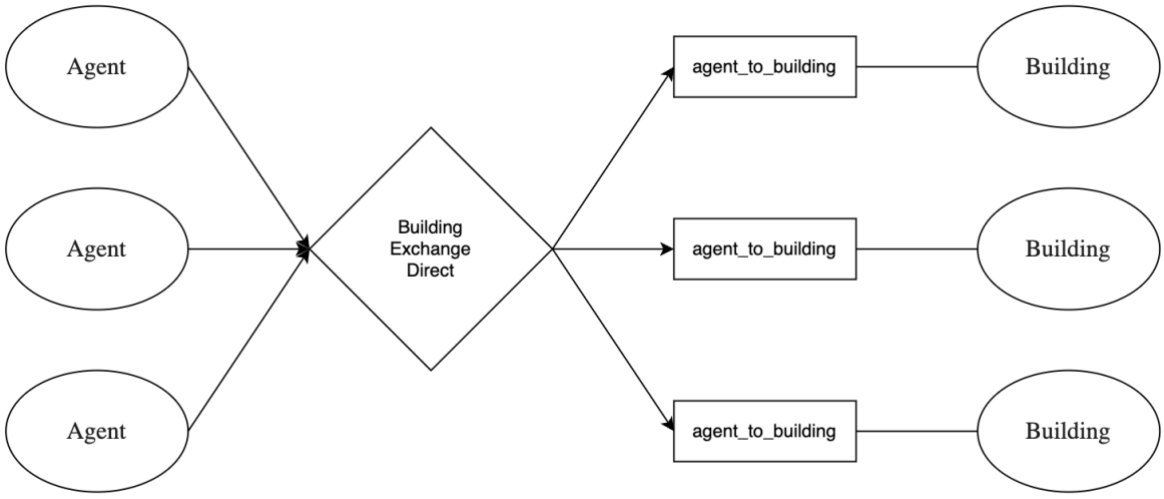


Logic for "on-the-fly" building connection:

- *Buildings to Agents*:
  - Buildings publish status updates to building_announce_exchange.
  - Messages are broadcast to all queues bound to this exchange, including agent_queue_{UUID} for each agent.

When a building connects, it sends information about its current status to ALL agents.

Figure 3. Adding new Agent



When a new Agent connects, it first sends a request to all buildings to inquire about their status, then receives booking information from other agents, and only after that begins accepting requests from clients.

Tests

1. A customer must be able to request a list of all buildings

```
Conference Room Booking System
1. Request list of buildings
2. Request your reservations
3. Exit
Choose an option: 1
Available Buildings and Rooms:
- Building B rooms: [201, 202, 203]
- Building C rooms: [301, 302, 303]
- Building A rooms: [101, 102, 103]
```

2. A customer can book one or more conference rooms in a building.

```
Available Buildings and Rooms:
- Building B rooms: [201, 202, 203]
- Building C rooms: [301, 302, 303]
- Building A rooms: [101, 102, 103]
1. Book rooms
2. Exit
Choose an option: 1
Enter building name: Building B
Enter room numbers separated by commas: 201, 202
```

3. When the rooms are available, the customer receives a unique reservation number.

```
1. Book rooms
2. Exit
Choose an option: 1
Enter building name: Building B
Enter room numbers separated by commas: 201, 202
Are you sure you want to book room(s) [201, 202] in building Building B? Confirmation number: 1f89971a-d7c4-4458-bcc5-a96128292829
```

4. If they are not available, the customer will also receive a message that this is the case.

```
Are you sure you want to book room(s) [201, 202] in building Building B? Confirmation number: 5cf8a408-3f1d-4619-9010-a175deab36c6
1. Confirm booking
2. Exit
Choose an option: 1
Error: Requested rooms are no longer available.
```

5. A reservation must be confirmed by the customer with the reservation number before it is final

```
Are you sure you want to book room(s) [201, 202] in building Building B? Confirmation number: 1f89971a-d7c4-4458-bcc5-a96128292829
1. Confirm booking
2. Exit
Choose an option: 1
Booking with confirmation number 1f89971a-d7c4-4458-bcc5-a96128292829 has been successfully confirmed.
```

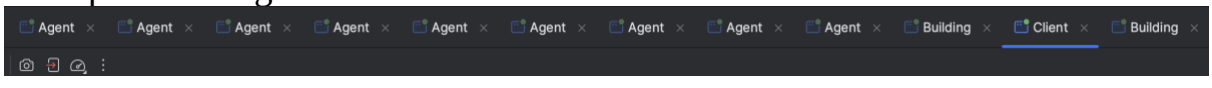
6. A customer must be able to cancel an existing reservation. This can be done by means of the reservation number.

```
Conference Room Booking System
1. Request list of buildings
2. Request your reservations
3. Exit
Choose an option: 2
Your reservations:
- Reservation confirmation number: 1f89971a-d7c4-4458-bcc5-a96128292829
1. Cancel a reservation
2. Exit
Choose an option: 1
Enter the reservation confirmation number you want to cancel: 1f89971a-d7c4-4458-bcc5-a96128292829
Reservation with confirmation number 1f89971a-d7c4-4458-bcc5-a96128292829 has been successfully cancelled.
```

7. New buildings must be able to be connected to the system on-the-fly.

```
Available Buildings and Rooms:
- Building O rooms: [101, 102, 103]
- Building B rooms: [203]
- Building R rooms: [201, 202, 203]
- Building C rooms: [301, 302, 303]
- Building A rooms: [101, 102, 103]
- Building D rooms: [301, 302, 303]
```

8. Multiple rental agents must be able to exist at the same time.



9. The system must neatly catch incorrect situations. For example, confirming a reservation with an unknown reservation number should result in a neat error message.

```
1. Cancel a reservation
2. Exit
Choose an option: 1
Enter the reservation confirmation number you want to cancel: adsffdsfgsdffsdfsdfgsfg
Invalid reservation number. Please enter a valid reservation confirmation number.
Enter the reservation confirmation number you want to cancel:
```

Conclusion

The implemented booking system for ConferenceRent.com effectively meets the specified requirements, demonstrating scalability, robustness, and efficient inter-process communication using RabbitMQ.

Key achievements include:

- *Scalable Architecture*: Supports multiple clients, rental agents, and buildings, allowing the system to handle increased load and growth.
- *Robust Communication*: Utilizes RabbitMQ exchanges and queues to facilitate reliable messaging between distributed components.
- *Data Consistency*: Agents synchronize booking data to maintain a consistent state across the system, ensuring accurate availability information.
- *Dynamic Component Management*: Buildings can be added or removed at runtime, and the system adapts seamlessly.
- *User-Friendly Client Interface*: Clients interact with the system without needing to know the underlying complexity, receiving prompt and accurate responses.