Fault Tolerant Server Algorithm Explanation

Server.java implements a normal server which sends an OK message periodically.
FaultyServer.java implements a faulty server which sends an OK message at an arbitrary time.
FaultTolerantServer.java implements a 1-fault tolerant server based on replication.
The code *Client.java* can be paired with any one of the server codes to receive messages and count the number of received messages.

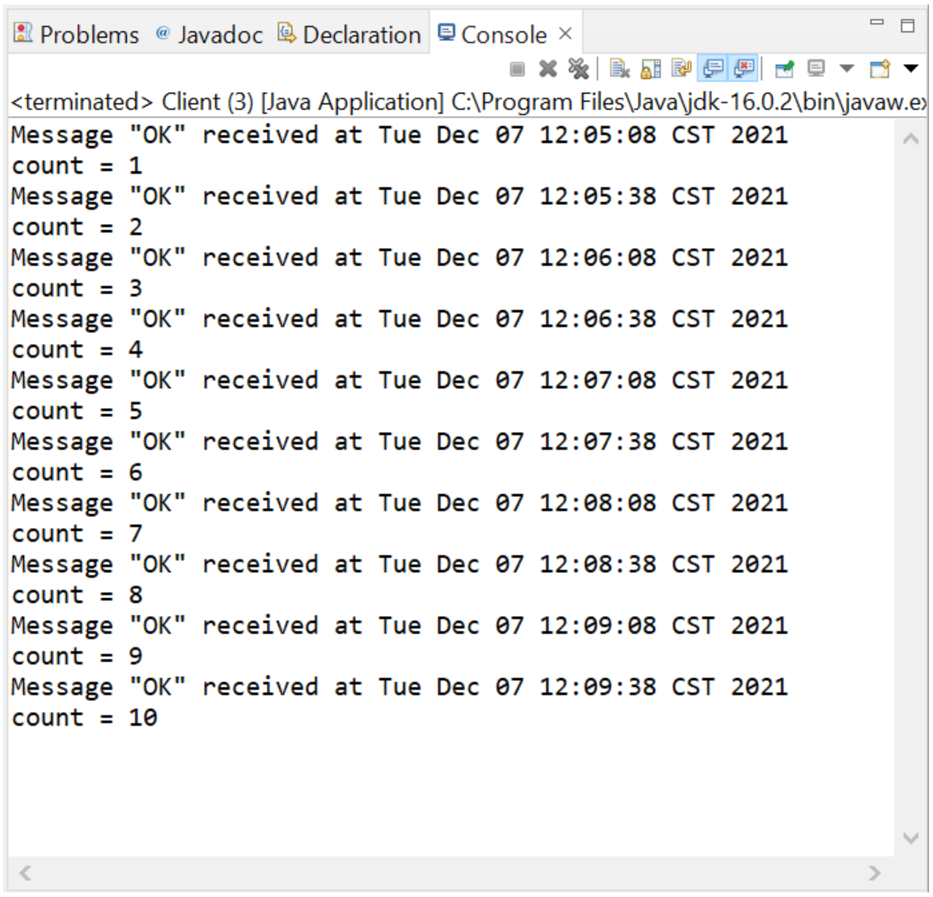
To implement 1-fault tolerant server, 3 server threads are required, because when a fault occurs in one thread, we need to reference the other two threads to make a decision. The basic idea of the algorithm is to send an OK message to the client only if more than half of the threads agree to send a message.

The fault-tolerant server code creates three sub-server threads among which a randomly selected one implements a faulty server and the other two normal servers. Each sub-server thread decides when to send a message independently. When it is time to send a message, each thread votes for sending a message by updating its own vote value in a vote box which is accessible by all threads. Then it tries to send the message by calling a *send* function.

The *send* function first checks the vote box to find out how many threads have voted to send a message, and only if at least two votes are found does the function actually send an OK message to the client. After sending the message, it clears the vote box for the next round.

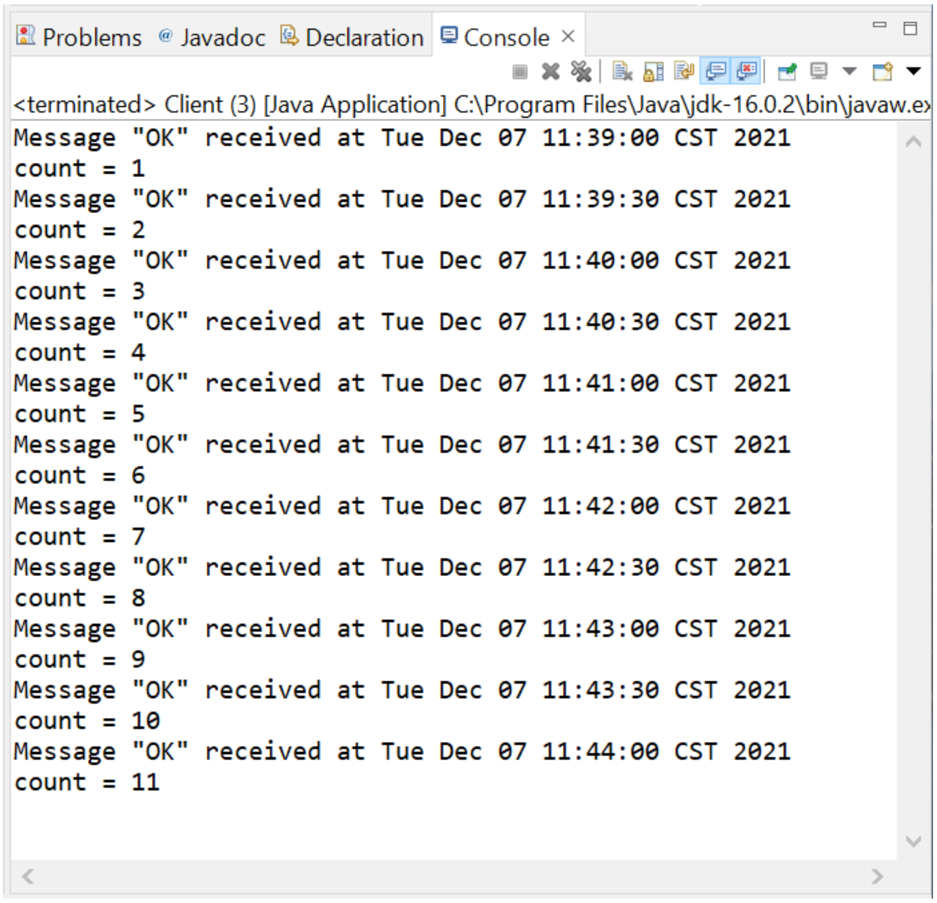
This *send* function is synchronized so that only one thread can access the function at a time. Hence, if multiple threads call the function at the same time, they call the function one by one, and because the function clears the vote box once it is called, only the first caller has a chance to actually send the message. This ensures that only one message is sent even if multiple threads try to send a message at the same time.

The following Fig. 1 shows that the client correctly receives an OK message from the fault-tolerant server every 30 seconds, which is the same as result of running the normal server (Fig. 2). The result of running the faulty server is shown in Fig. 3, where the client receives an OK message at an arbitrary time.



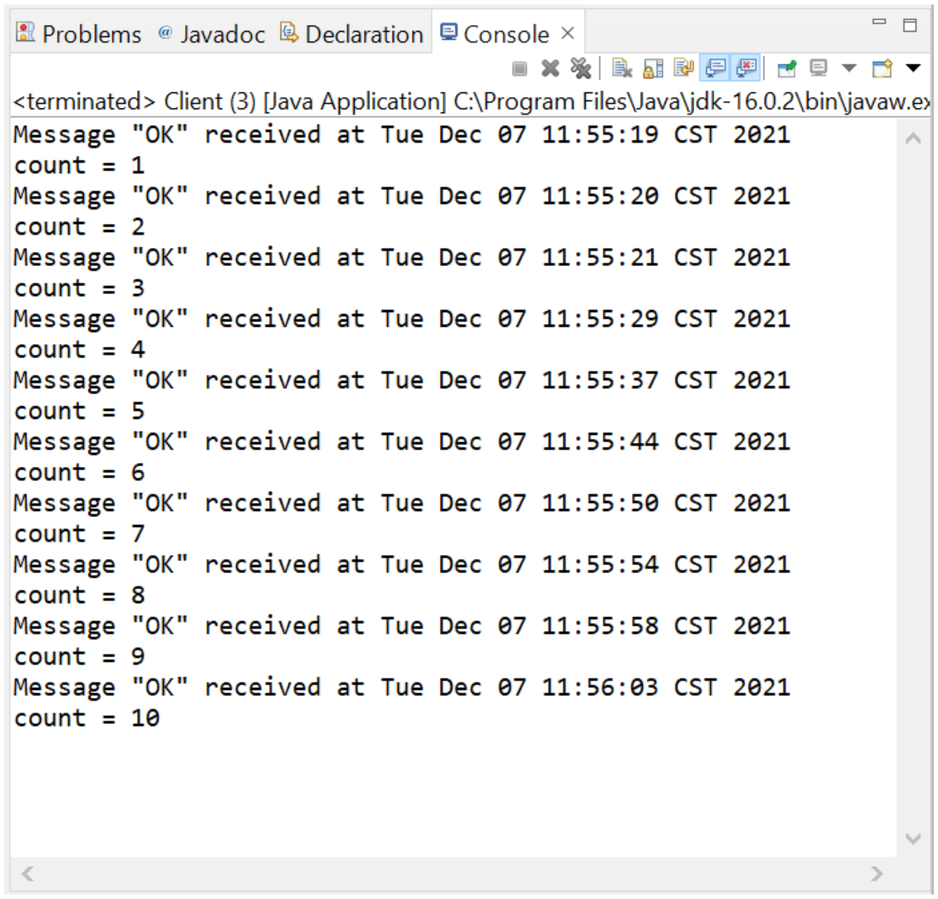
```
<terminated> Client (3) [Java Application] C:\Program Files\Java\jdk-16.0.2\bin\javaw.ex
Message "OK" received at Tue Dec 07 12:05:08 CST 2021
count = 1
Message "OK" received at Tue Dec 07 12:05:38 CST 2021
count = 2
Message "OK" received at Tue Dec 07 12:06:08 CST 2021
count = 3
Message "OK" received at Tue Dec 07 12:06:38 CST 2021
count = 4
Message "OK" received at Tue Dec 07 12:07:08 CST 2021
count = 5
Message "OK" received at Tue Dec 07 12:07:38 CST 2021
count = 6
Message "OK" received at Tue Dec 07 12:08:08 CST 2021
count = 7
Message "OK" received at Tue Dec 07 12:08:38 CST 2021
count = 8
Message "OK" received at Tue Dec 07 12:09:08 CST 2021
count = 9
Message "OK" received at Tue Dec 07 12:09:38 CST 2021
count = 10
```

Fig. 1 Client receiving messages from fault-tolerant server



```
<terminated> Client (3) [Java Application] C:\Program Files\Java\jdk-16.0.2\bin\javaw.e
Message "OK" received at Tue Dec 07 11:39:00 CST 2021
count = 1
Message "OK" received at Tue Dec 07 11:39:30 CST 2021
count = 2
Message "OK" received at Tue Dec 07 11:40:00 CST 2021
count = 3
Message "OK" received at Tue Dec 07 11:40:30 CST 2021
count = 4
Message "OK" received at Tue Dec 07 11:41:00 CST 2021
count = 5
Message "OK" received at Tue Dec 07 11:41:30 CST 2021
count = 6
Message "OK" received at Tue Dec 07 11:42:00 CST 2021
count = 7
Message "OK" received at Tue Dec 07 11:42:30 CST 2021
count = 8
Message "OK" received at Tue Dec 07 11:43:00 CST 2021
count = 9
Message "OK" received at Tue Dec 07 11:43:30 CST 2021
count = 10
Message "OK" received at Tue Dec 07 11:44:00 CST 2021
count = 11
```

Fig. 2 Client receiving messages from normal server



```
<terminated> Client (3) [Java Application] C:\Program Files\Java\jdk-16.0.2\bin\javaw.ex
Message "OK" received at Tue Dec 07 11:55:19 CST 2021
count = 1
Message "OK" received at Tue Dec 07 11:55:20 CST 2021
count = 2
Message "OK" received at Tue Dec 07 11:55:21 CST 2021
count = 3
Message "OK" received at Tue Dec 07 11:55:29 CST 2021
count = 4
Message "OK" received at Tue Dec 07 11:55:37 CST 2021
count = 5
Message "OK" received at Tue Dec 07 11:55:44 CST 2021
count = 6
Message "OK" received at Tue Dec 07 11:55:50 CST 2021
count = 7
Message "OK" received at Tue Dec 07 11:55:54 CST 2021
count = 8
Message "OK" received at Tue Dec 07 11:55:58 CST 2021
count = 9
Message "OK" received at Tue Dec 07 11:56:03 CST 2021
count = 10
```

Fig. 3 Client receiving messages from faulty server