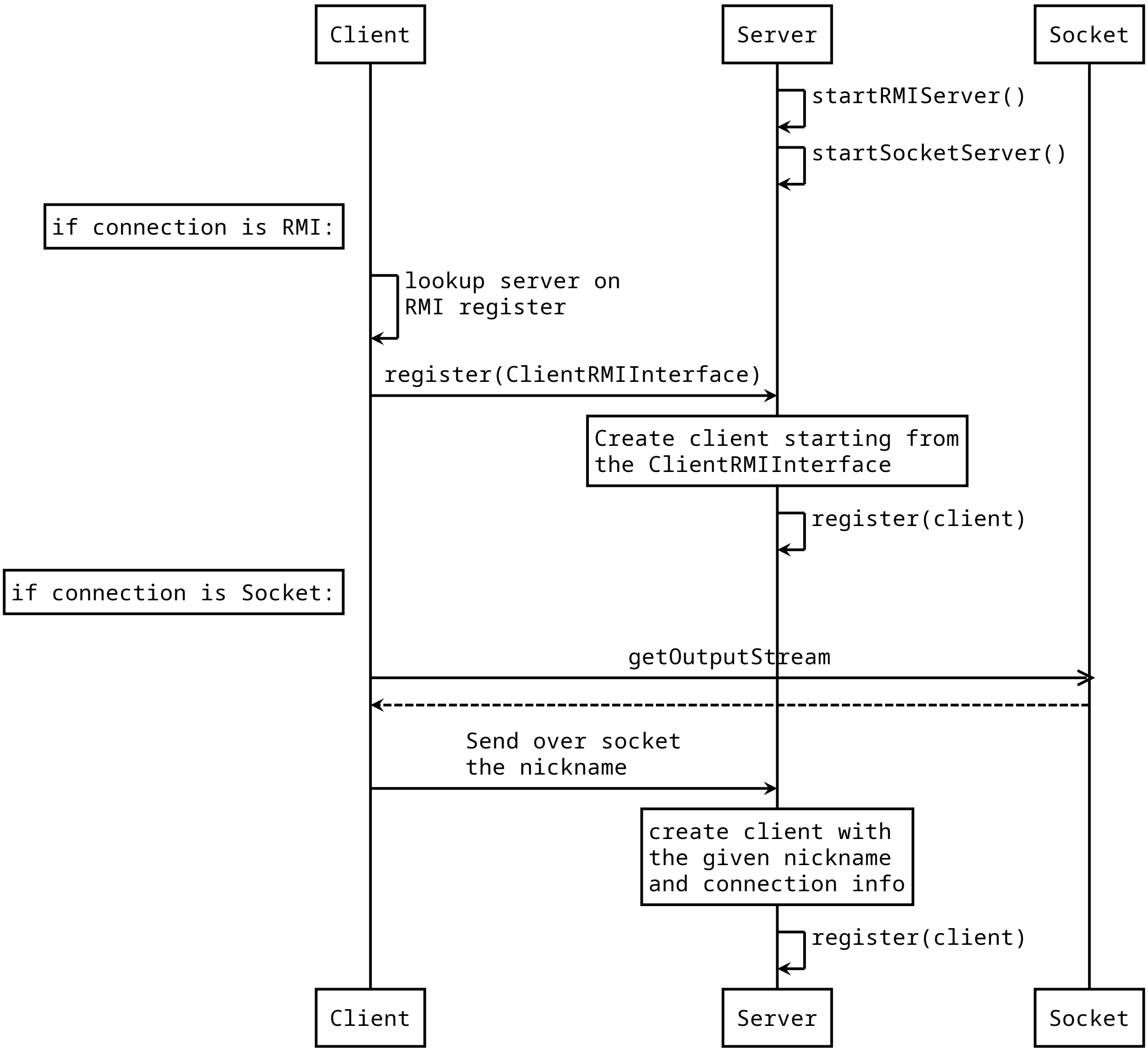
Network communication protocol

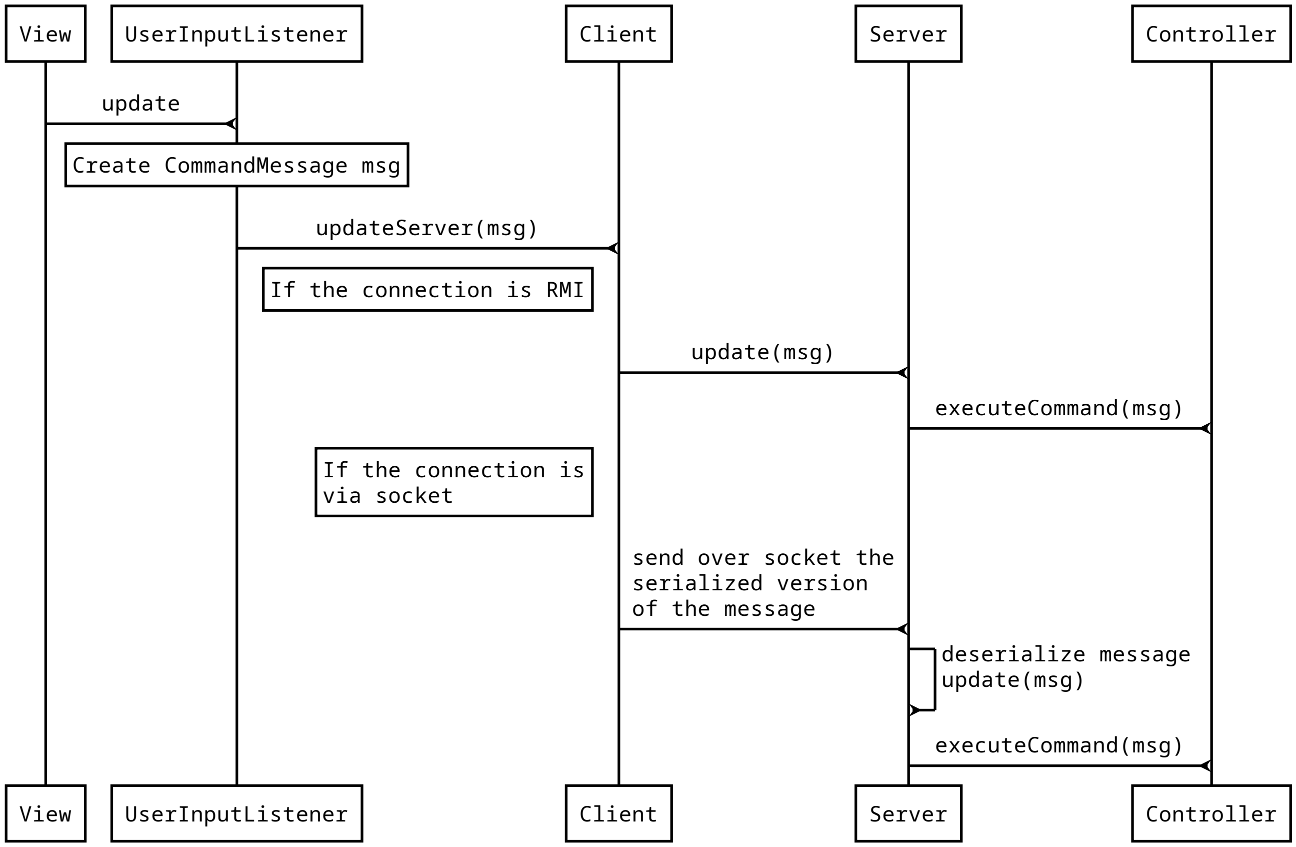
Sequence diagrams

Initialize connection


Client
Server
Socket
startRMIServer()
startSocketServer()
if connection is RMI:
lookup server on RMI register
register(ClientRMIInterface)
Create client starting from the ClientRMIInterface
register(client)
if connection is Socket:
getOutputStream
Send over socket the nickname
create client with the given nickname and connection info
register(client)
Client
Server
Socket

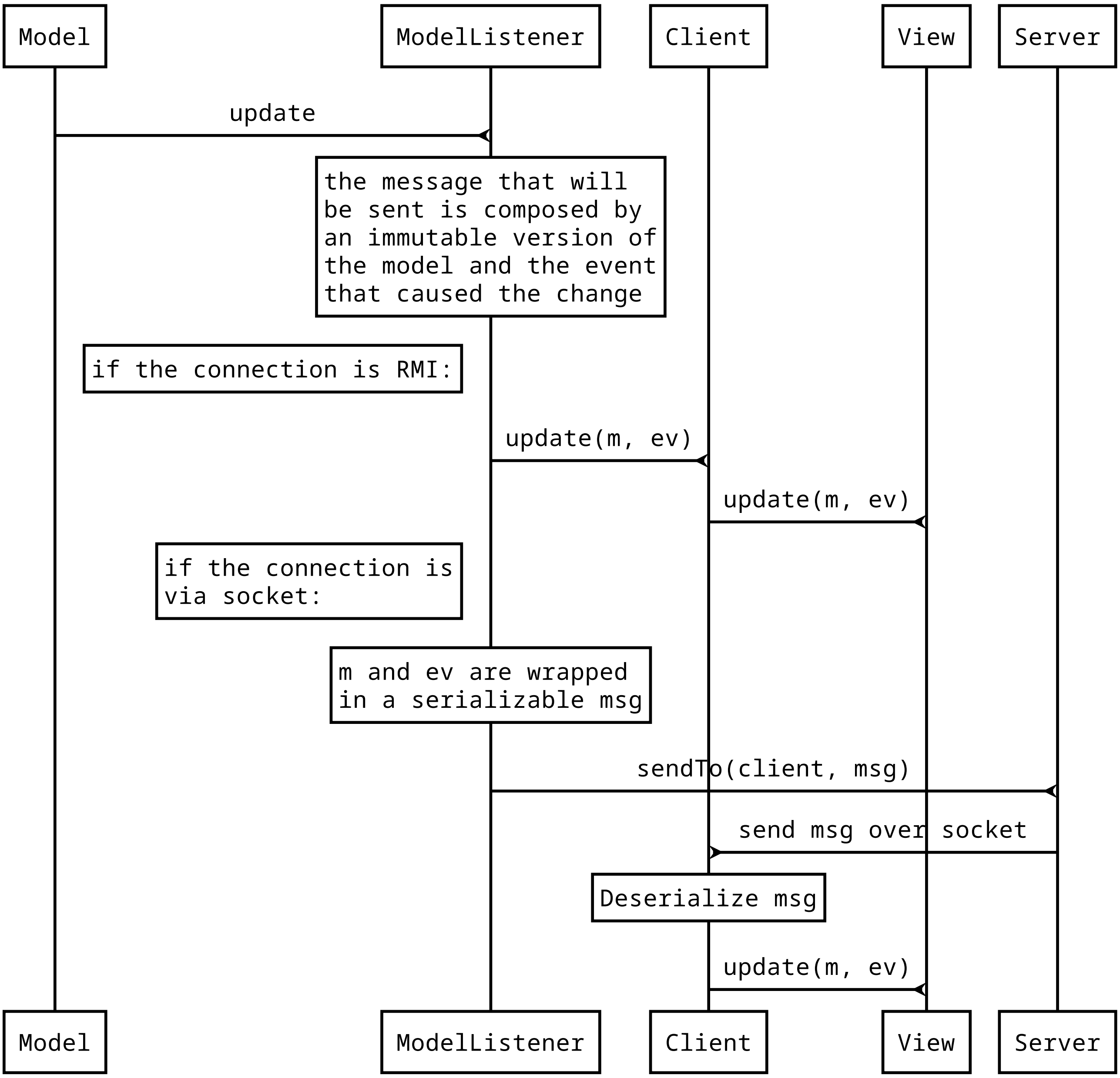
⚠️ Note that the to register the Client using RMI, the `register` method is called passing the `ClientRMIInterface` it implements and not the `Client` object, due to the way RMI works. (https://www.oracle.com/java/technologies/javase/remote-method-invocation-distributed-computing.html#3) Only then, the server creates a Client and calls the `register` method on itself.

View update - send message from one client to the server



Model update - send message from the server to all the registered clients

When a change occurs in the Model every subscribed `ModelListener` is notified and its update method called, starting the following interaction.

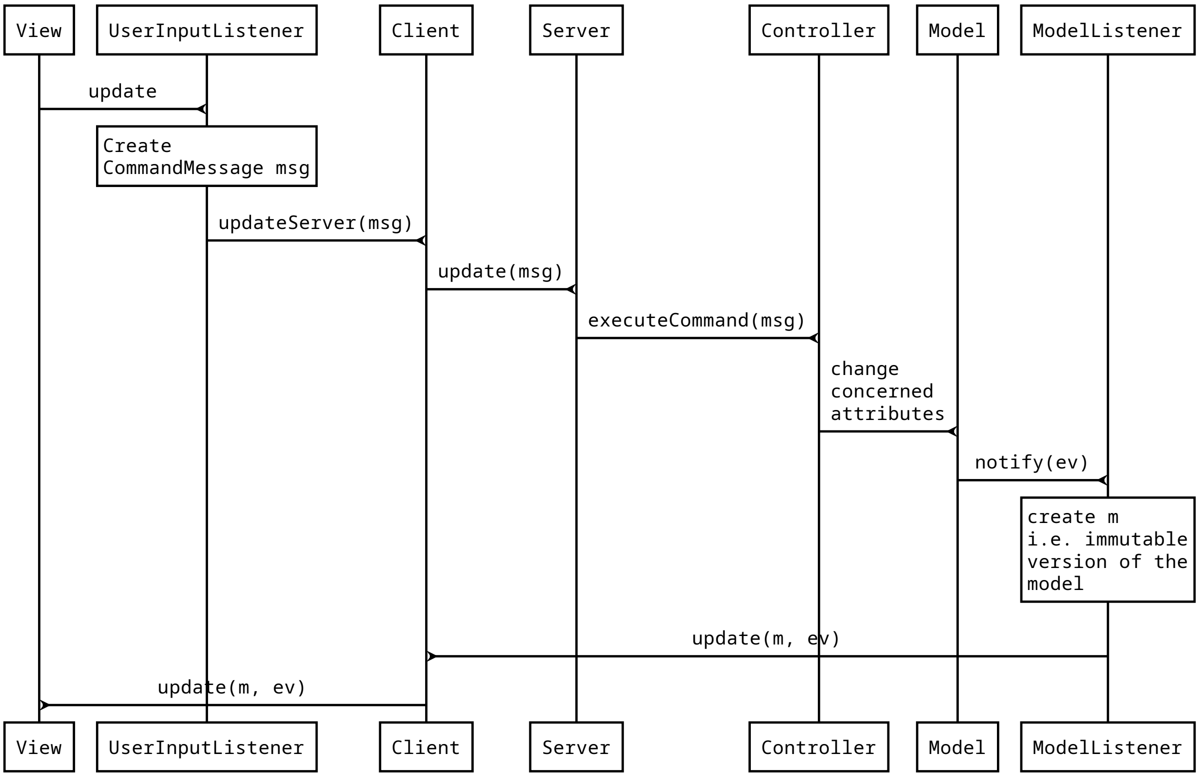

Model
ModelListener
Client
View
Server
update
the message that will be sent is composed by an immutable version of the model and the event that caused the change
if the connection is RMI:
update(m, ev)
update(m, ev)
if the connection is via socket:
m and ev are wrapped in a serializable msg
sendTo(client, msg)
send msg over socket
Deserialize msg
update(m, ev)
Model
ModelListener
Client
View
Server

- **RMI:** `ModelListener` is directly calling the `Client`'s remote `update` method
- **Sockets:** `ModelListener` calls `server.sendTo()` which is responsible for the communication via socket.

> ⚠️ This flow is the same also when an error occurs. In such cases the event parameter will point it out to the clients. This has been done both to better stick to MVC paradigm, and also to be able to possibly create a logger as a ModelListener able to save every change that occurs in the model.

Example: full chain of communication for a RMI client

In order to better understand the whole flow here is a diagram describing the interaction of an RMI client trying to perform an action.


View
UserInputListener
Client
Server
Controller
Model
ModelListener
update
Create
CommandMessage msg
updateServer(msg)
update(msg)
executeCommand(msg)
change
concerned
attributes
notify(ev)
create m
i.e. immutable
version of the
model
update(m, ev)
update(m, ev)
View
UserInputListener
Client
Server
Controller
Model
ModelListener