Exercises week 10

Last update: 2021/11/08

Goal of the exercises

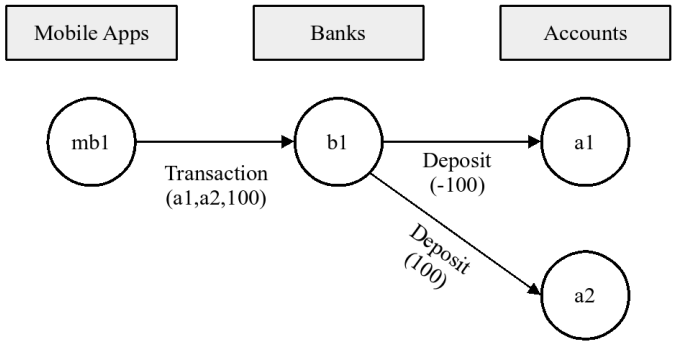
The goals of this week's exercises are:

- Design an Actor system.
- Design the communication protocol between actors.
- Implement the design in Akka.

Exercise 10.1 Your task in this exercise is to implement a Mobile Payment system using Akka. The system consists of three types of actors:

- *Mobile App*: These actors send transactions to the bank corresponding to mobile payments.
- *Bank*: These actors are responsible for executing the transactions received from the mobile app. That is, substracting the money from the payer account and adding it to the payee account.
- *Account*: This actor type models a single bank account. It contains the balance of the account. Also, banks should be able to send positive deposits and negative deposits (withdrawals) in the account.

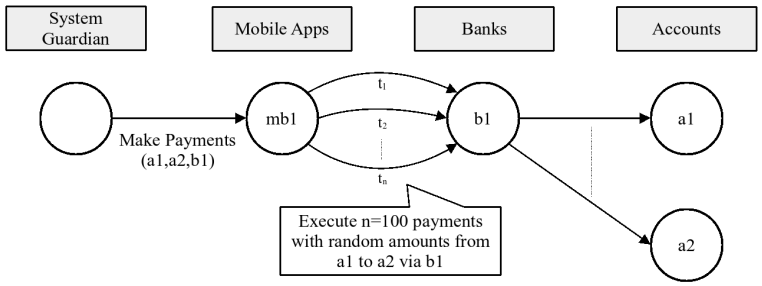
The directory `code-exercises/week10exercises` contains a source code skeleton for the system that you might find helpful.



The figure above shows an example of the behaviour. In this example, there is a mobile app, $mb1$, a bank, $b1$, and two accounts, $a1$ and $a2$. The arrow from $mb1$ to $b1$ models $mb1$ sending a transaction to $b1$ indicating to transfer 100 DKK from $a1$ to $a2$. Likewise, the arrows from $b1$ to $a1$ and $a2$ model the sending of deposits to the corresponding accounts to realise the transaction—the negative deposit can be seen as a withdrawal.

Mandatory

1. Implement the Bank and Account actors, including the messages and interaction to make deposits.
2. Implement the Mobile App actor, and define the messages mobile apps and banks exchange to perform transactions.
3. In Akka, it is recommended to have a top-level actor for the actors system that spawns the initial actors, and initiates the systems—often we refer to this actor as *guardian*. Your task here is to implement a guardian actor that starts 2 mobile apps, 2 banks, and 2 accounts. The guardian must be able to send a message to mobile app actors to execute a set of payments between 2 accounts in a specified bank. Finally, the guardian must send two payment messages: 1) from $a1$ to $a2$ via $b1$, and 2) from $a2$ to $a1$ via $b2$.



4. Modify the mobile app actor so that, when it receives a make payments message from the guardian, it sends 100 transactions between the specified accounts and bank with of random amount. Hint: You may use `Random::ints` to generate a stream of random integers. The figure above illustrates the computation.
5. Create a Main class that starts the actor system and sends the kickoff message that starts the computation to the guardian.

Challenging

6. Modify the system above so that Account actors are associated with a bank, and ensure that negative deposits are only executed if they are sent by that bank. In other words, only the account's bank is allowed to withdraw money. Note that positive deposits may be received from any bank.
7. Modify the system so that an account's balance cannot be below 0. If an account actor receives a negative deposit that reduces the balance below 0, the deposit should be rejected. In such a case, the bank should be informed and the positive deposit for the payee should not be performed.