UML Diagrams

Seth Hamrick

The UML activity and sequence diagrams both describe a “withdraw money” use case. The activity diagram provides this information in a more flowchart like format. Meanwhile the sequence diagram captures the information exchange more clearly. The activity diagram displays two decision nodes, the first to verify the pin and the second to verify the amount. The sequence diagram shows where and what information is passed during these points. The user messages are directly received by the ATM and then if information from the bank is required the ATM will message the bank directly to receive this.

I believe that the above description of the two diagrams is a perfect segway into a major deficiency within them. I believe that the communication that occurs during the pin verification should be mimicked on the sequence diagram to verify the amount as well. To dispense the cash to the user the ATM must verify two things. First, the ATM must verify that it has the amount of money requested. Second, the ATM must verify that the user has this amount of money in their account as well. For the ATM to do this it will need to communicate with the bank again to verify that the user account total is that or more than the amount requested. Another area that I see these diagrams may be enhanced slightly is the handling of an incorrect pin number. The current diagram sends the system immediately to the end of the use case, which likely leads to the removal of the card and the user to restart the process. I would imagine it is quite common that users may hit an incorrect key accidentally. In this event, if it is not a security concern, then I believe it may be easier to return the user to the beginning of the use case and allow the user another attempt to enter and verify the pin. This will increase usability as the process may continue even if the user is to make a mistake with their first attempt.

The deficiency I have selected to display will be the first relating to amount verification.

Diagram

Description automatically generated