Programmation système

Problems

Problem 1

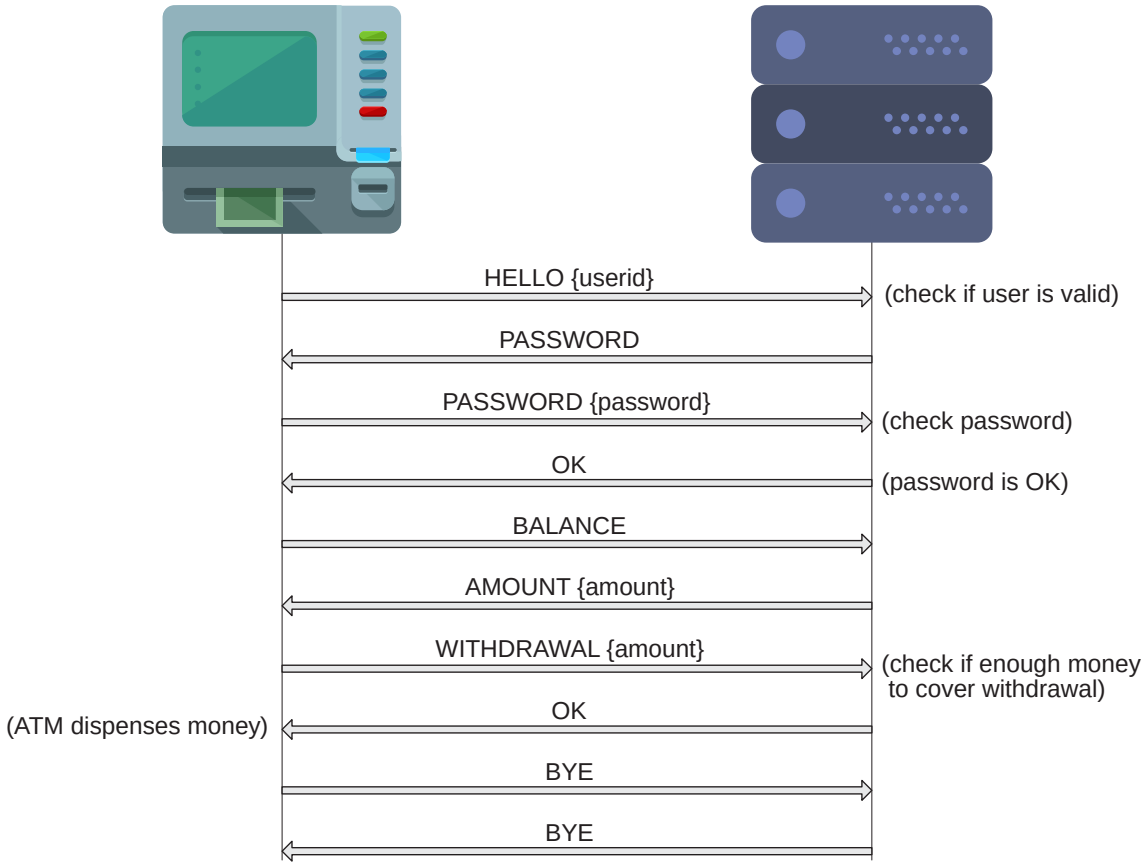
- Message from ATM to Server

Message name	**Purpose**
HELLO {userid}	Let server know that there is a card in the ATM machine ATM card transmists userID to the server
PASSWORD {password}	User enters PIN, which is sent to the server
BALANCE	User requests balance
WITHDRAWAL {amount}	User asks to withdraw money
BYE	User is all done

- Message from Server from ATM

Message name	**Purpose**
PASSWORD	Ask user for PIN (password)
OK	Last requested operation (PASSWORD, WITHDRAWAL) is OK
ERROR	Last requested operation (PASSWORD, WITHDRAWAL) is ERROR
AMOUNT {amount}	Sent in response to BALANCE request
BYE	User is done, display welcome screen at ATM

- Diagram of protocol: Simple withdrawal with no error



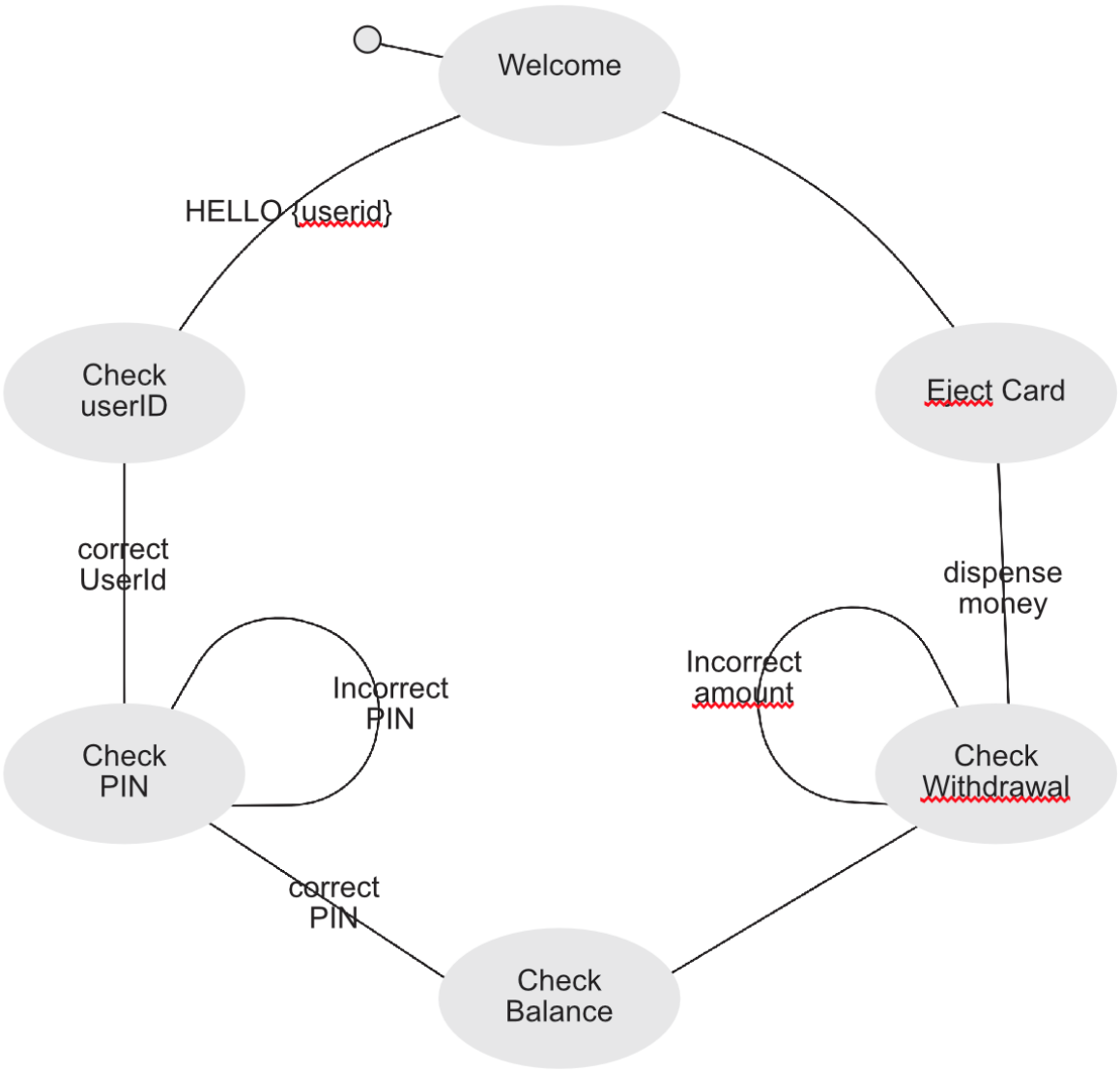
- Assumptions made by the protocol

Connection oriented: secure “connection” required between client and server

Reliable transport: data cannot be lost or received in wrong order

Flow control: the total amount of data transferred is small

- FSM



Problem 2

Client program: A piece of software that accesses a service made available by a server.
Server program: It's a service that shares information to its clients
No, it's the other way around. The client requests and receives services from the server.

A networking program usually has two programs, each running on a different host, communicating with each other. The program that initiates the communication is the client. Typically, the client program requests and receives services from the server program.

Problem 3

At first, the communication goes through all the layer. But then the legal departments and the presidents start talking to each other directly without going through the engineer department.

In the OSI protocol model, physical communication between peers takes place only in the lowest layer, not in every layer.

Problem 4

Byte stream =? Message stream

A reliable byte stream is a communication protocol that sends a continuous flow of data.

A reliable message stream is a communication protocol that sends distinct chunks of data

Problem 5

Negotiation has to do with getting both sides to agree on some parameters or values to be used during the communication. Maximum packet size is one example, but there are many others.

Problem 6

Same:

- Layers : network, transport, application
- Transport severice can provide a reliable end-to-end byte stream.

Difference:

- TCP/IP makes no assumptions about what happens above the level of a network session (part of OSI Layer 5), while OSI defines several more layers of standardized functions
- TCP/IP makes no prescriptions as to the link layers below IP, where OSI specifies two.
- OSI follows vertical approach as TCP/IP follows horizontal approach
- OSI has both connection oriented and connectionless service in the network layer

Problem 7

- Data link Layer
- Network layer

Problem 8

- The protocols of a network are fairly complex. Designing them in layers makes their implementation more manageable.
- Layering of protocols provides well-defined interfaces between the layers, so that a change in one layer does not affect an adjacent layer. That is the various functionalists can be partitioned and implemented independently, so that each one can be changed as technology improves without the other ones being affected. For example, a change in a routing algorithm of a network control program should not after affect the functions of message sequencing, which is located in another layer of the network architecture.

Problem 9

TCP is a connection-oriented protocol while UDP is a connectionless protocol.

Problem 10

In other words, TCP requires an active connection in order to deliver a message while UDP does not. If there is a connection failure during a TCP transaction, the server will request the lost part: there will not be any corruption. This is not the case with UDP: there might be corruption since when you send a message, there is no way to know if it will reach its destination.

Connection-oriented communication has 3 phases:

1. In the establishment phase a request is made to set up a connection
2. Only after this phase has been successfully completed can the data transfer phase be started and the data transported
3. Then comes the release phase.

Connectionless communication does not have these phases. It just sends the data.

Problem 11

In this case, frames encapsulate packets: indeed, when a packet reaches the data link layer, the entire packet including its header, the data, and everything else is being packed into the data field of the frame. Sometimes, this might not be possible if the packet does not fit. In this case, it has to be split into several smaller frames for a single packet.

Problem 12

Since each layer adds its own h-byte header, the total length of the message is:

$$Headers = n * h$$

$$Total = M + Headers$$

$$Total = M + n * h$$

Hence the fraction of the network bandwidth filled with headers is:

$$F = \frac{Headers}{Total}$$

$$F = \frac{n * h}{M + n * h}$$

Problem 13

Depending on the reliability of the network, it may be wise to use the first or the second approach. If the network tends to lose packets, it is better to chop the file into several packets so lost packets can be retransferred. However, if the network is reliable, it is more interesting to send the whole file without acknowledge each packet individually: this will help to save bandwidth. Unfortunately, in rare cases, if there has been a packet loss, the entire file will have to be resent.

Problem 14

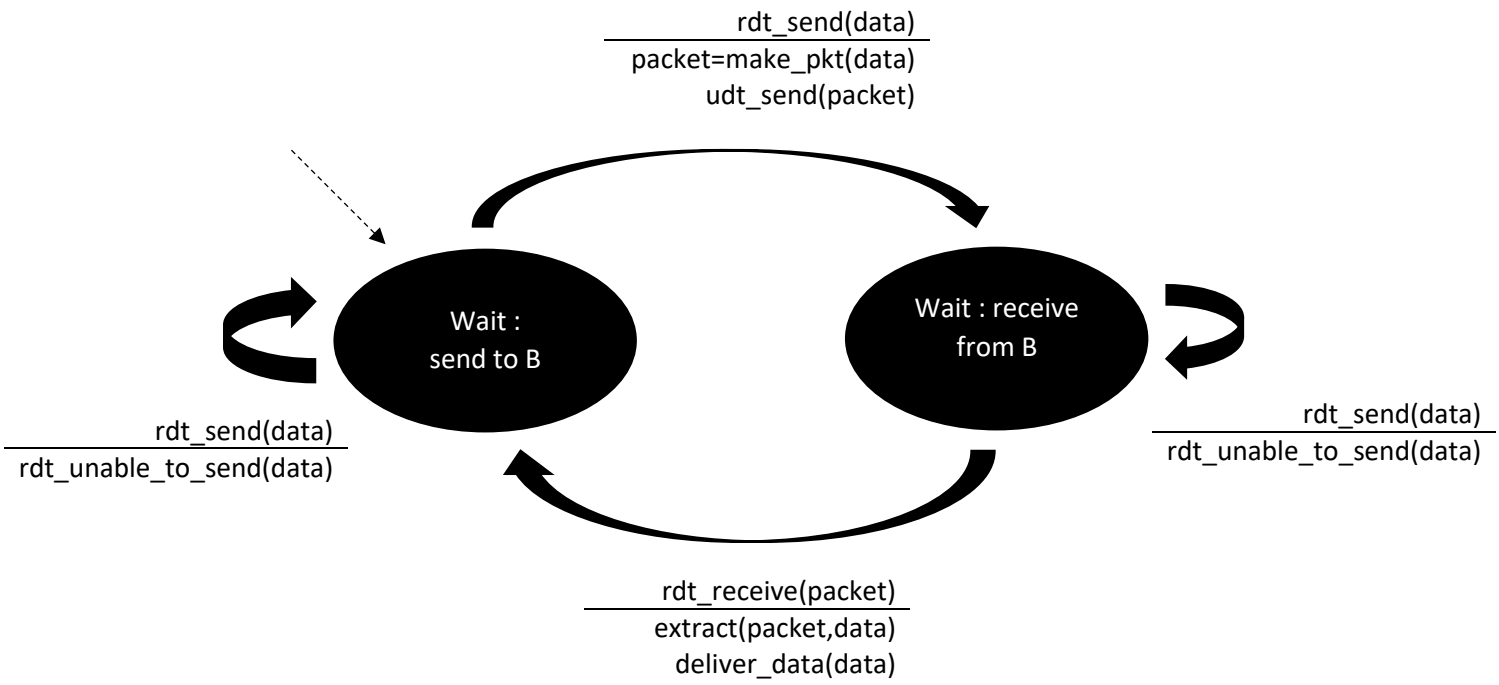
In the case of a NAK-only protocol, if the sender sends data only infrequently, he will not know if the current package has been received or not until he sends the next packet. Due to the low frequency of the transactions, it will take a long time to recover the first packet. In this case, the sender should use ACKs.

In other words: In a NAK only protocol, the loss of packet x is only detected by the receiver when packet x+1 is received. That is, the receivers receive x-1 and then x+1, only when x+1 is received does the receiver realize that x was missed. If there is a long delay between the transmission of x and the transmission of x+1, then it will be a long time until x can be recovered, under a NAK only protocol. On the other hand, if data is being sent often, then recovery under a NAK-only scheme could happen quickly. Moreover, if errors are infrequent, then NAKs are only occasionally sent (when needed), and ACK are never sent – a significant reduction in feedback in the NAK-only case over the ACK-only case.

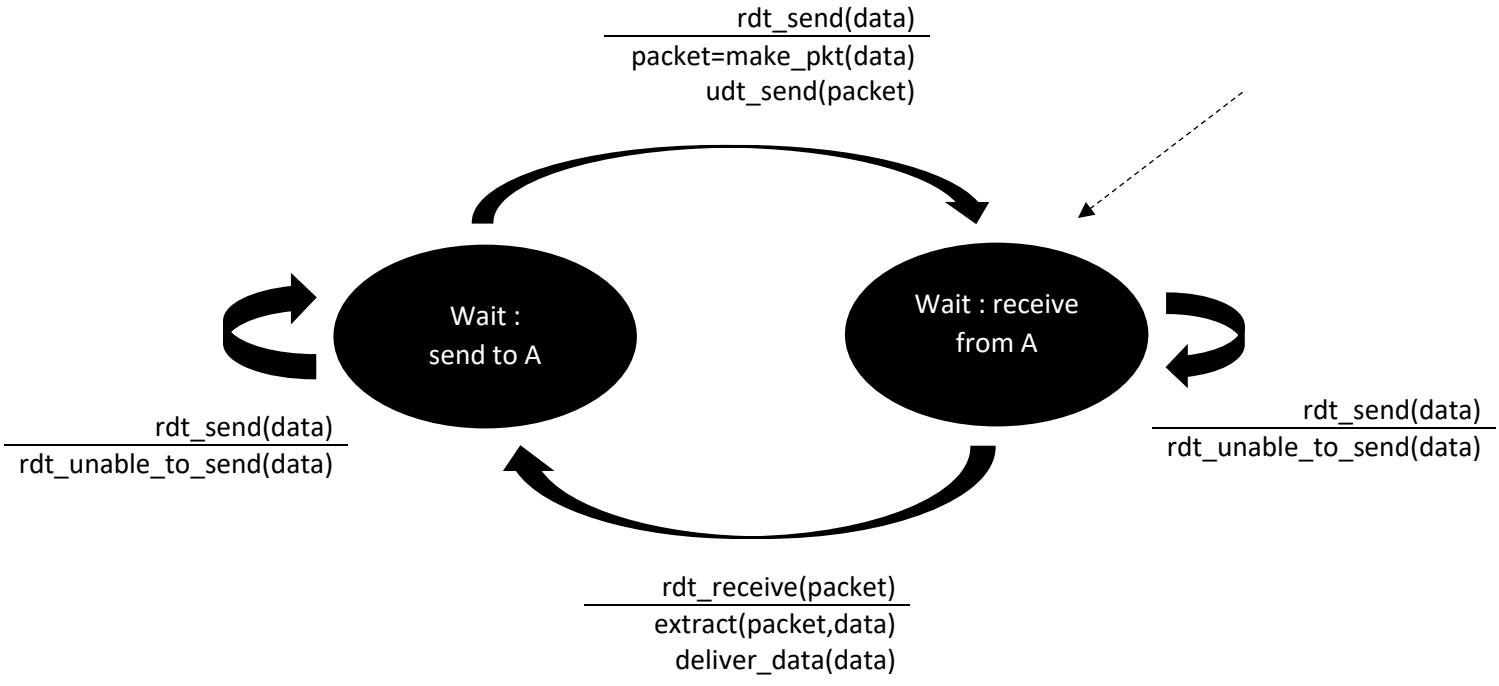
However, if the sender has a lot of data, it is interesting to use a NAK-only protocol since it will be significantly faster than a protocol that uses ACKs: since errors should be much less frequents than successes, there will be much less acknowledgments to be expected, thus a faster transaction.

Problem 15

Entity A:



Entity B:



Sometimes, speed is preferred over reliability; for video games or streaming an occasional packet loss is tolerable and preferred over lagging. In these cases, unreliable communication is preferred over reliable communication.