1. a) The reachability graph of the net:

$$\begin{tikzcd}
 & & m_0{:}\{s_0,r_0,d_0,a_0\} \arrow[dll, "\text{send}(m_1)"'] \arrow[drr, "\text{send}(m_2)"] & & \\
m_1{:}\{s_3,r_0,d_0,a_0\} \arrow[d, "\text{sdata}(m_1)"'] & & & & m_2{:}\{s_1,r_0,d_0,a_0\} \arrow[d, "\text{sdata}(m_2)"] \\
m_3{:}\{s_2,r_0,d_1,a_0\} \arrow[d, "\text{rdata}(m_1)"'] \arrow[drr, "\text{ldata}(m_1)"] & & & & m_4{:}\{s_2,r_0,d_2,a_0\} \arrow[d, "\text{rdata}(m_2)"] \arrow[dll, "\text{ldata}(m_2)"'] \\
m_5{:}\{s_2,r_3,d_0,a_0\} \arrow[drr, "\text{rec}(m_1)"'] & & m_6{:}\{s_2,r_0,d_0,a_0\} & & m_7{:}\{s_2,r_1,d_0,a_0\} \arrow[dll, "\text{rec}(m_2)"] \\
 & & m_8{:}\{s_2,r_2,d_0,a_0\} \arrow[d, "\text{sack}"] & & \\
 & & m_9{:}\{s_2,r_0,d_0,a_1\} \arrow[uu, bend left=60, "\text{lack}"] \arrow[uuuuu, bend left=90, "\text{rack}"] & &
\end{tikzcd}$$

b) It is possible that the receiver does not receive the sent message, because there is a deadlock in the reachability graph. The deadlock occurs in marking m_6 in the reachability graph. The deadlock happens if a message is lost (transitions ldata(m_1), ldata(m_2) or lack). For each message sent, at most one message is received.

c) The model can be simplified by discarding the message identities. The simplification can be achieved by removing all transitions related to the message m_2 (send(m_2), sdata(m_2), rdata(m_2), rec(m_2) and ldata(m_2)). After removing the transitions, places that are not connected to any transition can also be removed. Similarly, one can remove from the reachability graph arcs corresponding to transitions mentioned above. Markings m_2, m_4 and m_7 then become unreachable and can thus be removed.

2. The protocol no longer deadlocks, but it still does not work correctly. The receiver can not distinguish re-sent messages from new ones. There are paths in the reachability graph where the receiver gets arbitrarily many messages (transition rec), but the sender only has sent one message (transition send). An example of an erroneus path (only the fired transitions are presented): *send*, sdata, rdata, *rec*, sack, timeout, sdata, rdata, *rec*,. . . . In this path there are already two rec transitions, even though the send transition was fired only once. If acknowledgements are lost, it is possible that the receiver will get even more excess messages.

   If the transmission channel is reliable (transitions ldata and lack are removed), the problem does not disappear. The example path mentioned before is still possible. However, the amount ef excess messages that the receiver can get is now limited to one.

   The changes also created a deadlock to the system. The deadlock is reached if the transitions timeout, sdata, timeout are appended to the path mentioned before. Because acknowledgements can not be lost, the we end up in a situation where both the acknowledgement channel and the data channel are full. The receiver can not read messages from the data channel before it has sent the acknowledgement, and the sender can not read the acknowledgement before it has sent the message.

   The deadlock is a property of the *model*, and does not necessarily appear in a concrete (real) system. The small capacity of the channels is an abstraction made in the model. Therefore a deadlock occurring because of channels filling up is not necessarily an error in the concrete system.

   By adding the transition timeout, the size of reachability graph grows considerably. With lossy channels, the reachability graph has 34 markings and 89 arcs, and with reliable channels 33 markings and 56 arcs.