Stop-and-Wait Protocol

The sender has a control variable, which we call S (sender), that points to the only slot in the send window. The receiver has a control variable, which we call R (receiver), that points to the only slot in the receive window.

FSMs

Sender The sender is initially in the ready state, but it can move between the ready and blocking state. The variable S is initialized to 0.

❑ Ready State. When the sender is in this state, it is only waiting for one event to occur. If a request comes from the application, the sender creates a packet with the sequence number set to S. A copy of the packet is stored, and the packet is sent. The sender then starts the only timer. The sender then moves to the blocking state.

❑ Blocking State. When the sender is in this state, three events can occur:

1. If an error-free ACK arrives with ackNo related to the next packet to be sent, which means ackNo= (S + 1) modulo 2, then the timer is stopped. The window is slided, S = (S + 1) modulo 2. Finally, the sender moves to the ready state.

2. If a corrupted ACK or an error-free ACK with the ackNo ≠ (S + 1) modulo 2

arrives, the ACK is discarded.

3. If a time-out occurs, the sender resends the only outstanding packet and restarts the timer.

Receiver The receiver is always in the ready state. The variable R is initialized to 0.

Three events may occur:

1. If an error-free packet with seqNo = R arrives, the message in the packet is delivered to the application layer. The window then slides, R = (R + 1) modulo 2. Finally an ACK with ackNo = R is sent.

2. If an error-free packet with seqNo ≠ R arrives. The packet is discarded, but an ACK with ackNo = R is sent.

3. If a corrupted packet arrives, the packet is discarded.

Go-Back-N Protocol

Sender The sender starts in the ready state, but it can be in one of the two states after:

ready and blocking. The two variables are normally initialized to 0 (Sf = Sn = 0), but we

will see in the future chapters that some protocols in the TCP/IP protocol use a different

initialization.

❑ Ready State. Four events may occur when the sender is in ready state.

1. If a request comes from the application layer, the sender creates a packet with the sequence number set to Sn. A copy of the packet is stored, and the packet is sent. The sender also starts the only timer if it is not running. The value of Sn is now incremented, (Sn = Sn + 1) modulo 2m. If the window is full, Sn = (Sf + Ssize) modulo 2m, the sender goes to the blocking state.

2. If an error-free ACK arrives with ackNo related to one of the outstanding packets, the sender slides the window (set Sf = ackNo) and if all outstanding packets are acknowledged (ackNo = Sn), then the timer is stopped. If all outstanding packets are not acknowledged, the timer is restarted.

3. If a corrupted ACK or an error-free ACK with ackNo not related to the outstanding packet arrives, it is discarded.

4. If a time-out occurs, the sender resends all outstanding packet and restarts the timer.

❑ Blocking State. Three events may occur in this case:

1. If an error-free ACK arrives with ackNo related to one of the outstanding packets, the sender slides the window (set Sf = ackNo) and if all outstanding packets are acknowledged (ackNo = Sn), then the timer is stopped. If all outstanding packets are not acknowledged, the timer is restarted. The sender then moves to the ready state.

2. If a corrupted ACK or an error-free ACK with the ackNo not related to outstanding packets arrives, the ACK is discarded.

3. If a time-out occurs, the sender sends all outstanding packets and restarts the timer.

Receiver The receiver is always in the ready state. The only variable Rn is initialized to 0. Three events may occur:

1. If an error-free packet with seqNo = Rn arrives, the message in the packet is delivered to the application layer. The window then slides, Rn = (Rn + 1) modulo 2m. Finally an ACK is sent with acqNo = Rn.

2. If an error-free packet with seqNo outside the window arrives, the packet is discarded, but an ACK with ackNo = Rn is sent.

3. If a corrupted packet arrives, it is discarded.

Selective-Repeat Protocol

Sender The sender starts in the ready state, but later it can be in one of the two

states: ready or blocking. The following shows the events and the corresponding actions

in each state.

❑ Ready State. Four events may occur in this case:

1. If a request comes from the application layer, the sender creates a packet with the sequence number set to Sn. A copy of the packet is stored, and the packet is sent. If the timer is not running, the sender starts the timer. The value of Sn is now incremented, Sn = (Sn + 1) modulo 2m. If the window is full, Sn = (Sf + Ssize) modulo 2m, the sender goes to the blocking state.

2. If an error-free ACK arrives with ackNo related to one of the outstanding packets, that packet is marked as acknowledged. If the ackNo = Sf, the window slides to the right until the Sf points to the first unacknowledged packet (all consecutive acknowledged packets are now outside the window). If there are outstanding packets, restarts the timer; else, stops the timer.

3. If a corrupted ACK or an error-free ACK with ackNo not related to an outstanding packet arrives, it is discarded.

4. If a time-out occurs, the sender resends all unacknowledged packets in the win-

dow and restarts the timer.

❑ Blocking State. Three events may occur in this case:

1. If an error-free ACK arrives with ackNo related to one of the outstanding packets, that packet is marked as acknowledged. In addition, if the ackNo = Sf, the window is slid to the right until the Sf points to the first unacknowledged packet (all consecutive acknowledged packets are now outside the window). If the window has slid, the sender moves to the ready state.

2. If a corrupted ACK or an error-free ACK with the ackNo not related to outstanding packets arrives, the ACK is discarded.

3. If a time-out occurs, the sender resends all unacknowledged packets in the window and restarts the timer.

Receiver The receiver is always in the ready state. Three events may occur:

1. If an error-free packet with seqNo in the window arrives, the packet is stored and an ACK with ackNo = seqNo is sent. In addition, if the seqNo = Rn, then the packet and all previously arrived consecutive packets are delivered to the application layer and the window slides so that the Rn points to the first empty slot.

2. If an error-free packet with seqNo outside the window arrives, the packet is discarded, but an ACK with ackNo = Rn is returned to the sender. This is needed to let the sender slide its window if some ACKs related to packets with seqNo < Rn were lost.

3. If a corrupted packet arrives, the packet is discarded.