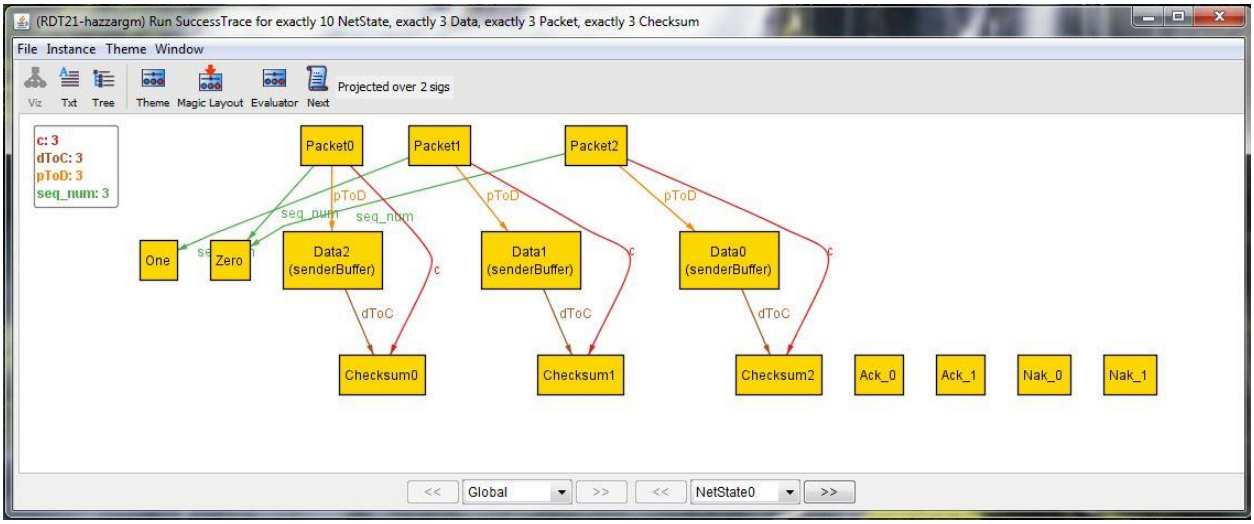
Team: Moore Hazzard
Members: Gordon Hazzard & Jordan Moore

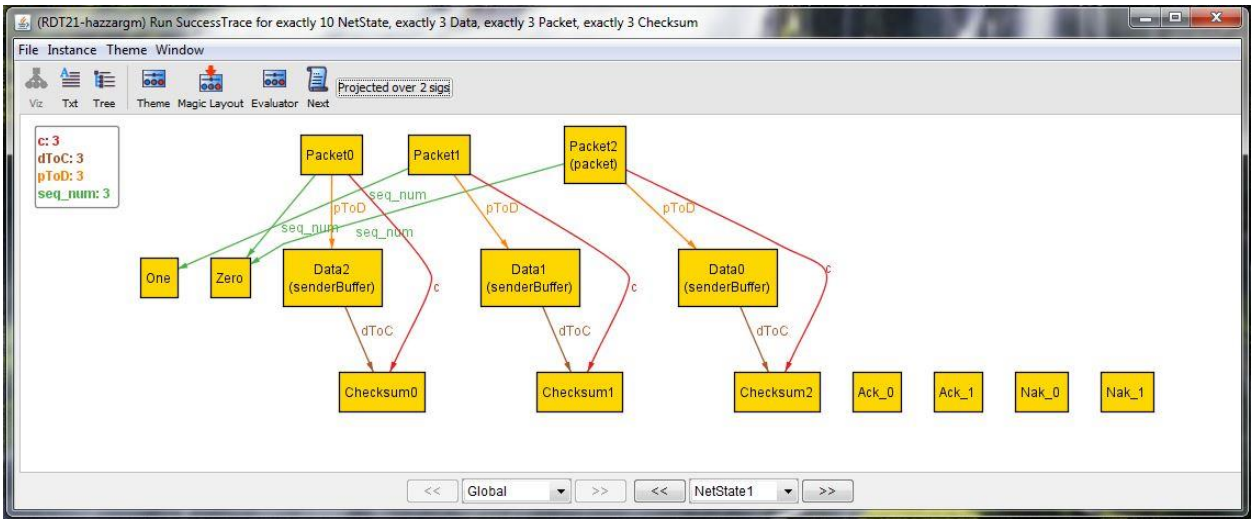
Sprint 3 – Checking RDT 2.1

Property 1: "It is possible to transmit all of the data in the sender's buffer to the receiver's buffer."

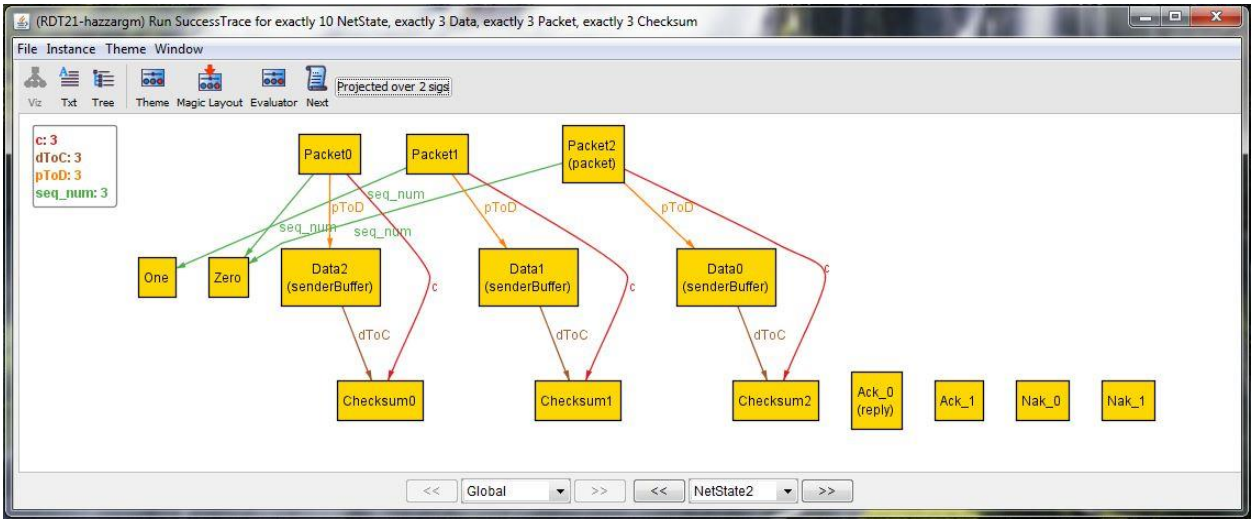
Below is a series of snapshots that confirms the first property. The images depict the transmission of 3 sets of Data, each with a unique Checksum, via 3 Packets. The packets are sent with alternating sequence numbers.



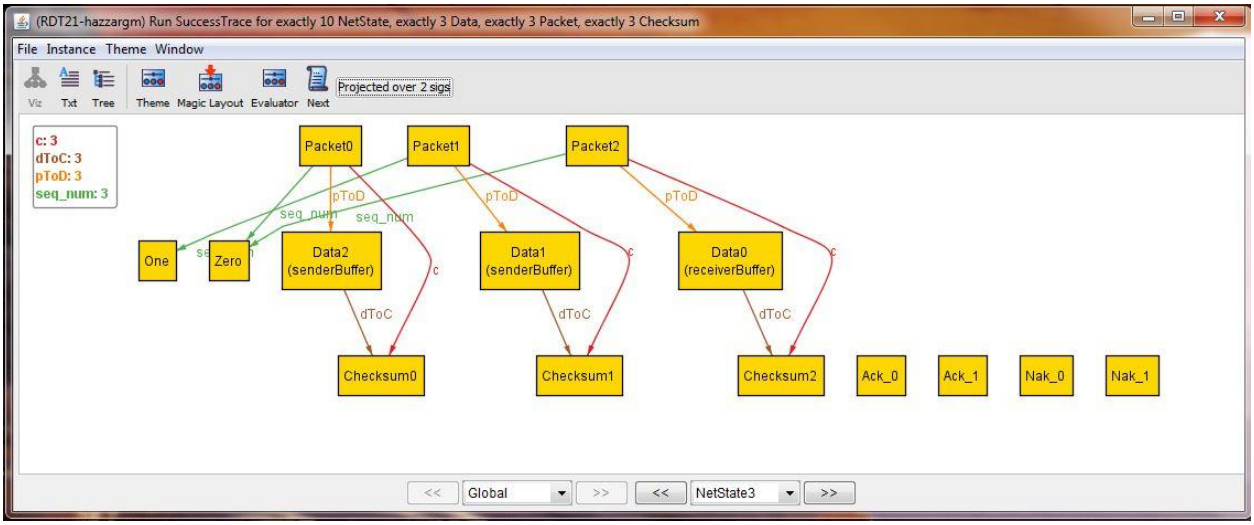
This is the initial state. All Data objects are in the senderBuffer and none are in the receiverBuffer. Additionally none of the Data have been packaged into a Packet yet. Both Data object's Checksum value has not been calculated nor has that Checksum been associated with that Data's packet yet. The future mappings of such relationships are only shown above via the Global signature which has knowledge of all States. Ack_0, Ack_1, Nak_0, and Nak_1 represent acknowledgements, and since nothing has yet been sent or received, none of them are associated with this state. The number indicates the sequence number that it corresponds to. Zero and One are the sequence numbers, and each packet points to a sequence number because it is a global property, though like the checksums, these relationships have not yet been established in the timeline.



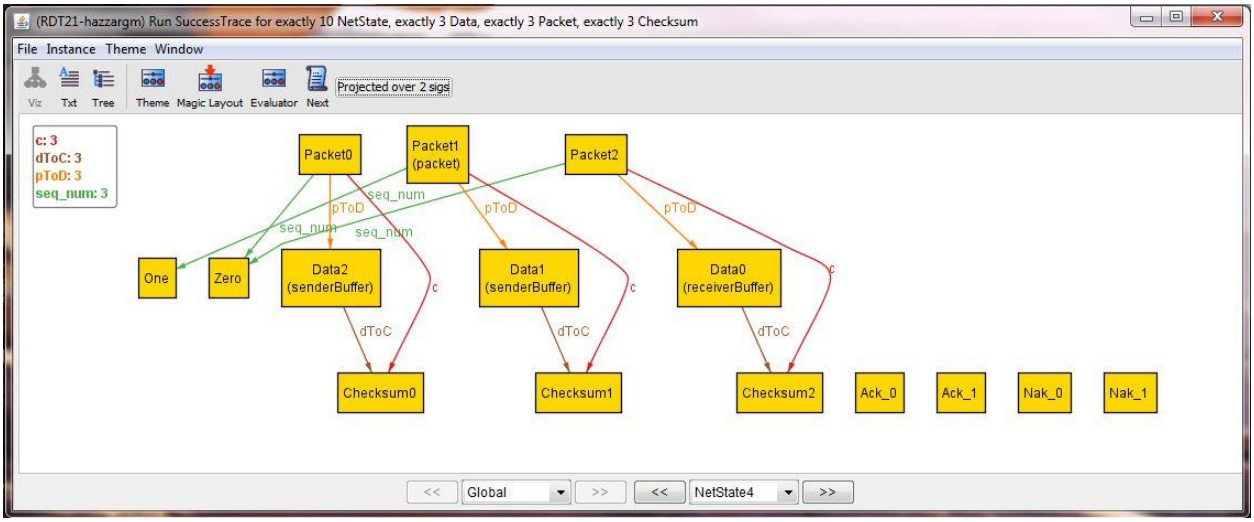
In the next state, Data0 has been made into a Packet, specifically the Packet2 which it is designated to map to in the Global signature. By now, its Checksum value would have been calculated and associated with the packet. This state shows that Data0 is in the process of being transmitted. Data0 will remain in the senderBuffer until an acknowledgement is received. Additionally, Packet2 has now been assigned the sequence number Zero.



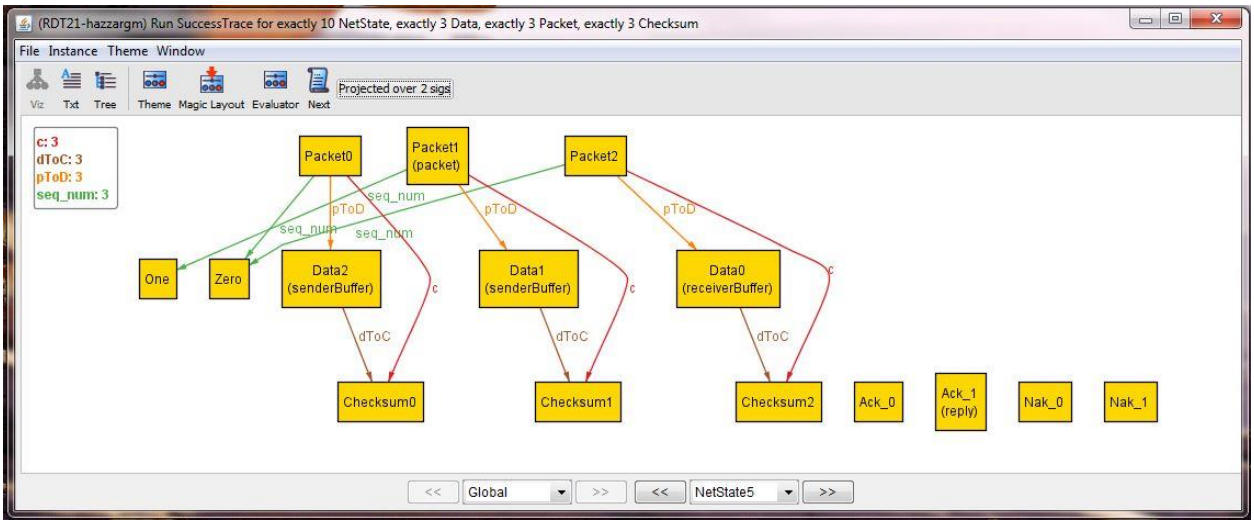
This state shows an acknowledgement through its ‘reply’ field which is Ack_0, meaning that Data0 and Packet2 are not corrupt (i.e. Data0 maps to Checksum1 and Packet1’s checksum field is also Checksum1). It also means that nothing went awry in the reply transmission because Ack_0 matches the sequence number Zero to which the packet mapped.



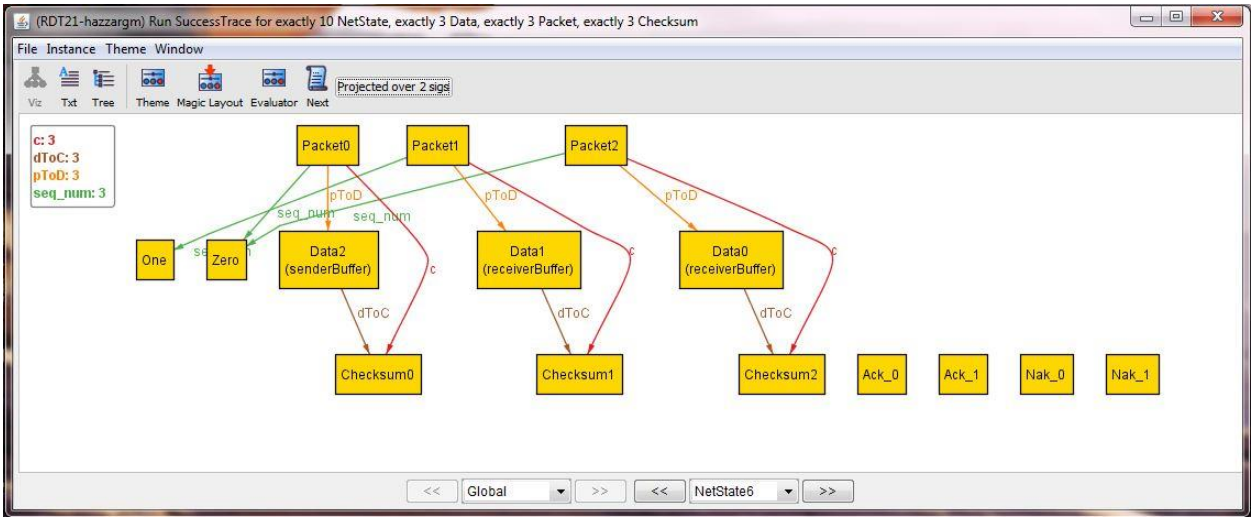
Next up is a receive state. Data0's transmission has just been successfully acknowledged in the previous state, so the receiver removes Data0 from its packet and places it in the receiverBuffer. The sender, having received the Ack_0 acknowledgement message, removes Data0 from the senderBuffer. There are no packets in transmission or acknowledgments in this state.



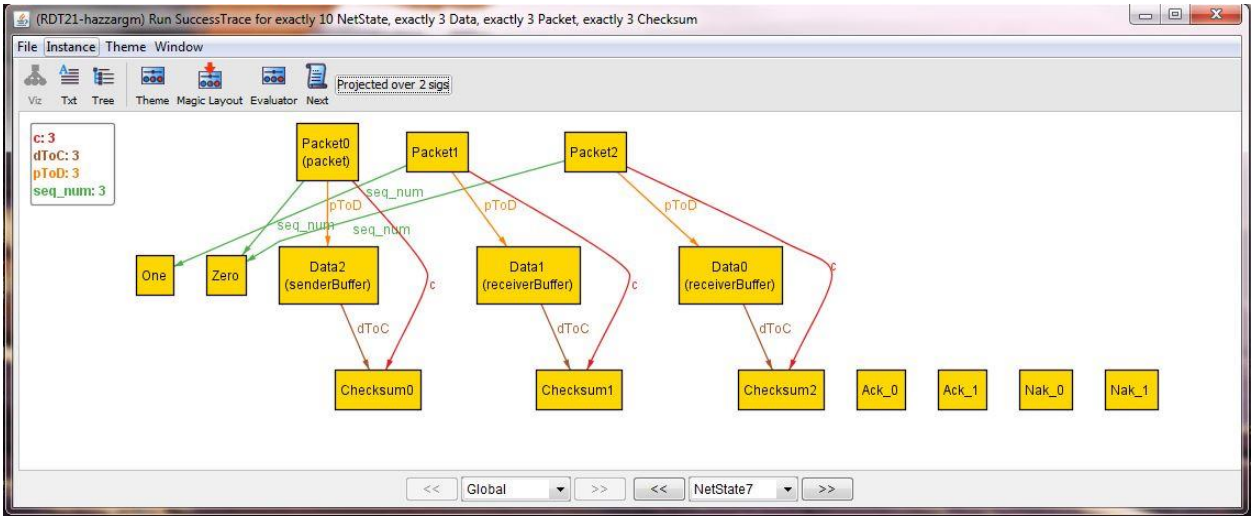
This state is nearly identical to NetState1 and shows the sending of Data1. The key difference is that Packet1 has a sequence number of One which is 'next up' in the alternation.



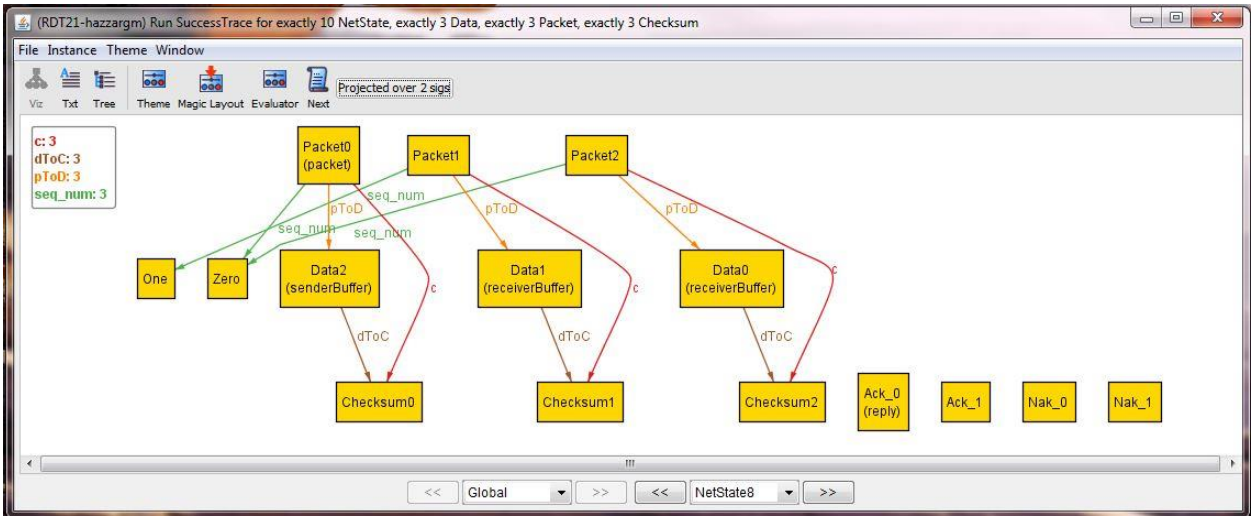
This state is nearly identical to NetState2 and shows the acknowledgement after receiving Data1.The reply, Ack_1, matches the sequence number of the packet that was sent which was One, and therefore indicates that everything is functioning as specified.



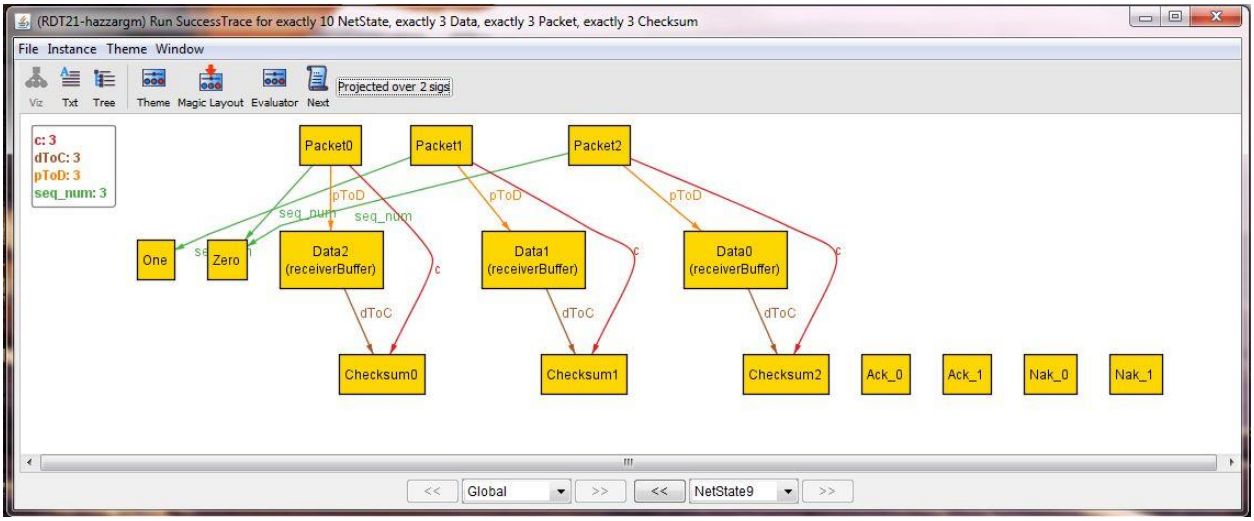
This state is identical to NetState3, except that Data1 is the object being delivered.



This state is semantically identical to NetState1, except Data2 is now being sent. Note that Packet0 has a sequence number of Zero, the next one up in the pattern of alternation.



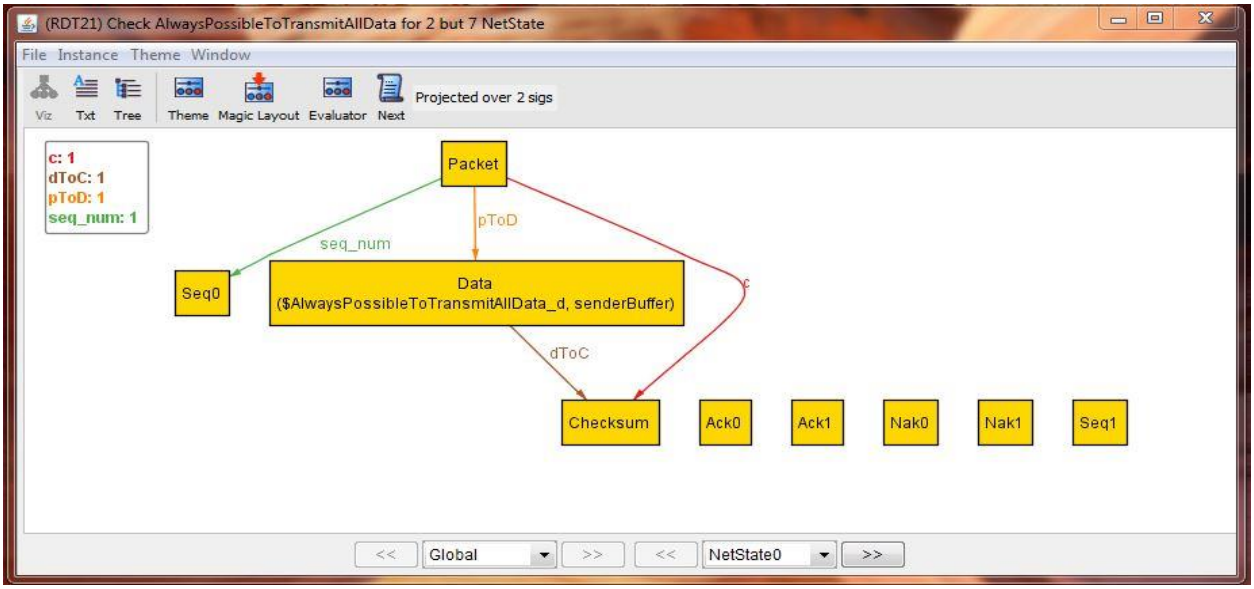
Semantically identical to NetState2, the Ack_0 reply matches the Zero sequence number of the packet.

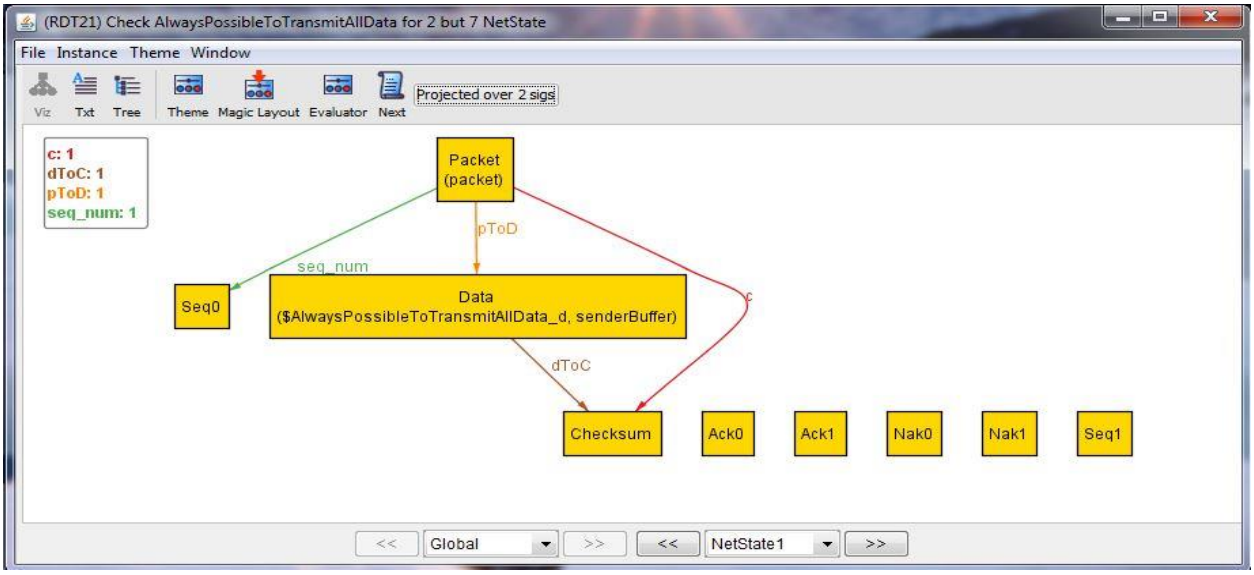


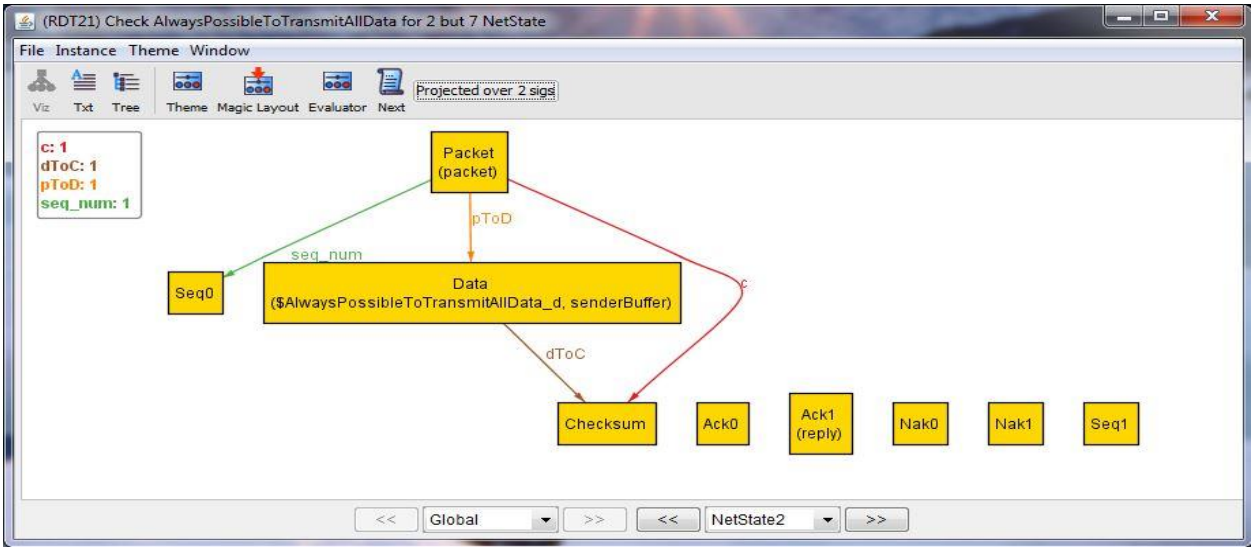
Finally, this state is semantically equivalent to NetState3 and shows the outcome of successfully delivery of Data2. This is also the end state since all Data has successfully be transmitted and is now in the receiverBuffer, no Data remain in the senderBuffer, and no extra packets/acknowledgement messages are floating around in transmission.

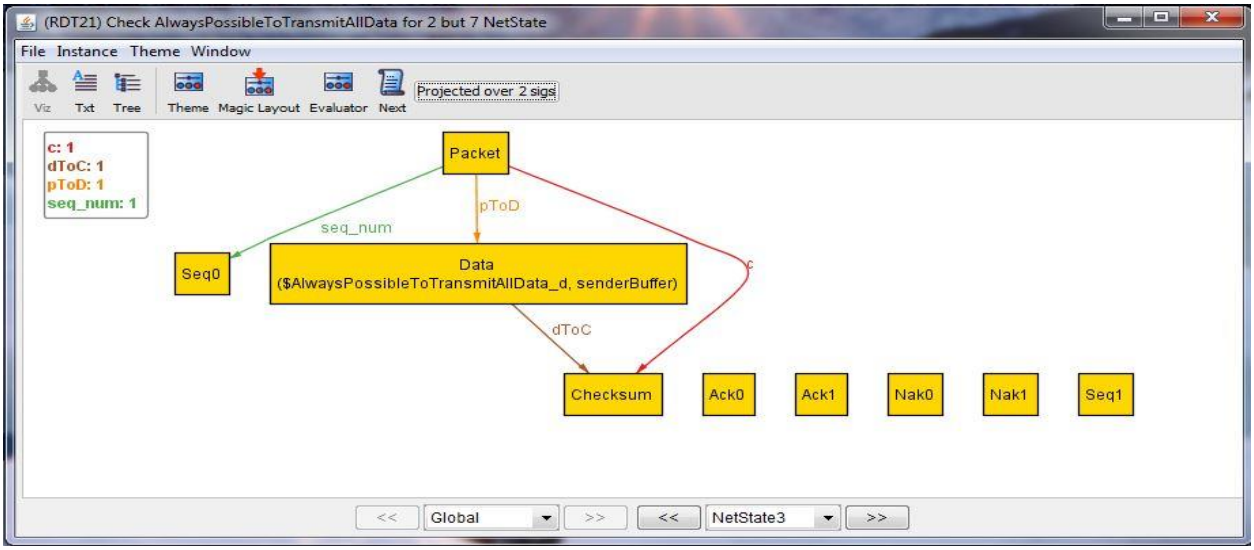
Property 2: "It is *always* possible to transmit all of the data in the sender's buffer to the receiver buffer."

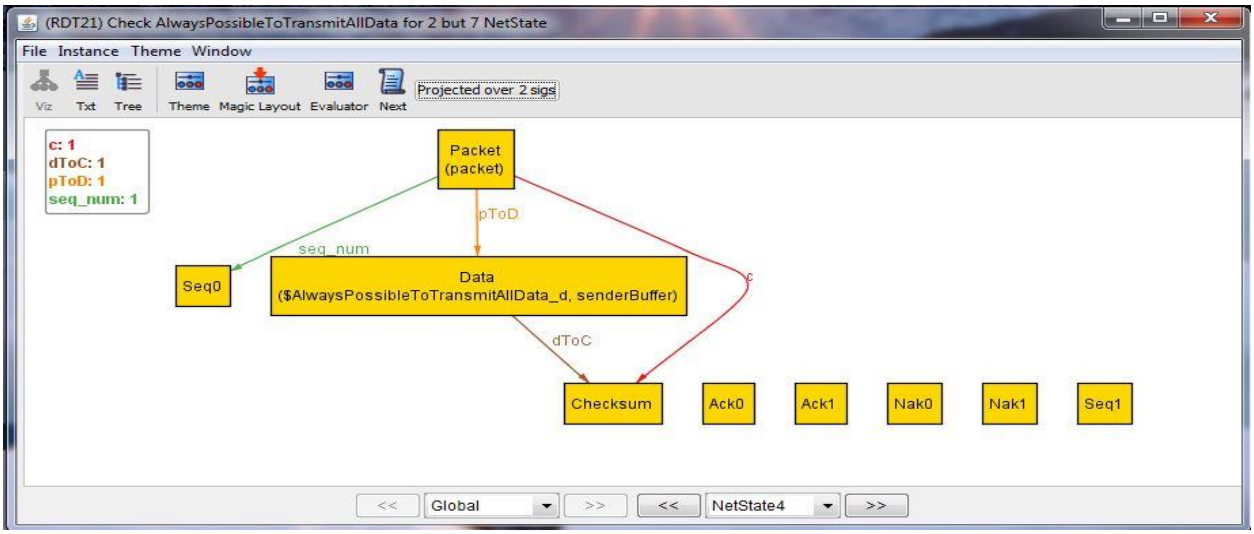
Below is a series of snapshots of a counterexample that refutes the claim of Property 2. Since RDT2.1 uses an unreliable channel were data and/or acknowledgement replies can become corrupted, there is a possible situation in where a certain packet or reply becomes corrupted repeatedly and its data is never successfully transmitted. This phenomenon is shown below as something gets garbled and results in a sequence mismatch. The sender keeps resending the packet with the sequence number Zero because it keeps getting an Ack_1 reply.

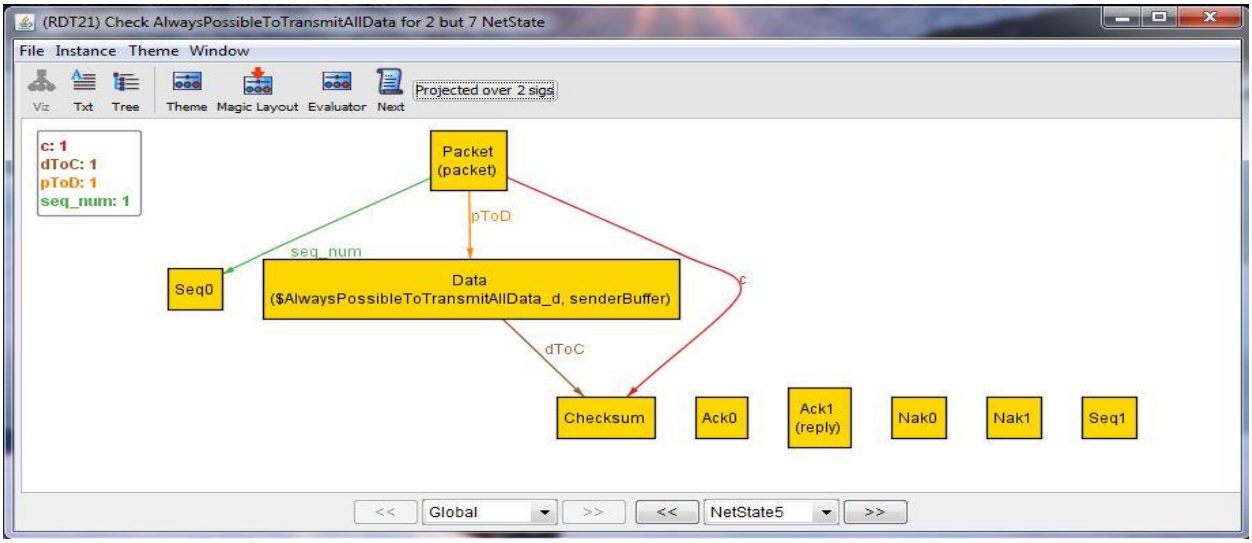


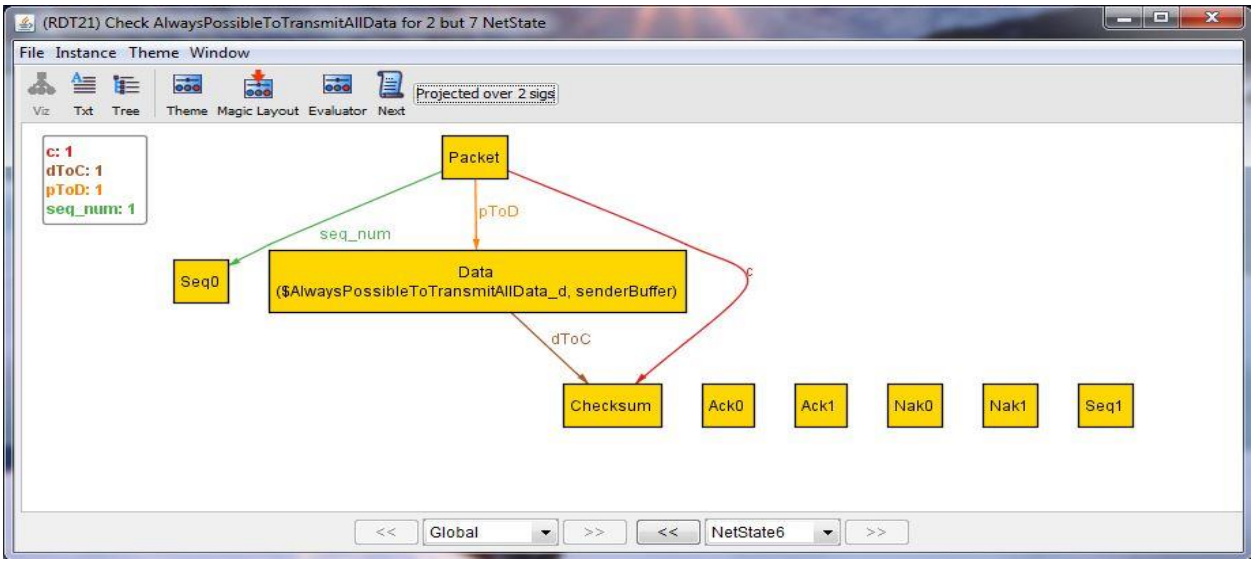

(RDT21) Check AlwaysPossibleToTransmitAllData for 2 but 7 NetState
File Instance Theme Window
Viz Txt Tree Theme Magic Layout Evaluator Next
Projected over 2 sigs
c: 1
dToC: 1
pToD: 1
seq_num: 1
Packet (packet)
pToD
seq_num
Seq0
Data ($AlwaysPossibleToTransmitAllData_d, senderBuffer)
c
dToC
Checksum
Ack0
Ack1
Nak0
Nak1
Seq1
<< Global >> << NetState1 >>


(RDT21) Check AlwaysPossibleToTransmitAllData for 2 but 7 NetState
File Instance Theme Window
Viz Txt Tree Theme Magic Layout Evaluator Next
Projected over 2 sigs
c: 1
dToC: 1
pToD: 1
seq_num: 1
Packet (packet)
pToD
seq_num
Seq0
Data ($AlwaysPossibleToTransmitAllData_d, senderBuffer)
c
dToC
Checksum
Ack0
Ack1 (reply)
Nak0
Nak1
Seq1
<< Global >> << NetState2 >>


(RDT21) Check AlwaysPossibleToTransmitAllData for 2 but 7 NetState
File Instance Theme Window
Viz Txt Tree Theme Magic Layout Evaluator Next
Projected over 2 sigs
c: 1
dToC: 1
pToD: 1
seq_num: 1
Packet
pToD
seq_num
Seq0
Data ($AlwaysPossibleToTransmitAllData_d, senderBuffer)
c
dToC
Checksum
Ack0
Ack1
Nak0
Nak1
Seq1
<< Global >> << NetState3 >>


(RDT21) Check AlwaysPossibleToTransmitAllData for 2 but 7 NetState
File Instance Theme Window
Viz Txt Tree Theme Magic Layout Evaluator Next
Projected over 2 sigs
c: 1
dToC: 1
pToD: 1
seq_num: 1
Packet
(packet)
pToD
seq_num
Seq0
Data
($AlwaysPossibleToTransmitAllData_d, senderBuffer)
c
dToC
Checksum
Ack0
Ack1
Nak0
Nak1
Seq1
<< Global >> << NetState4 >>


(RDT21) Check AlwaysPossibleToTransmitAllData for 2 but 7 NetState
File Instance Theme Window
Viz Txt Tree Theme Magic Layout Evaluator Next
Projected over 2 sigs
c: 1
dToC: 1
pToD: 1
seq_num: 1
Packet
(packet)
pToD
seq_num
Seq0
Data
($AlwaysPossibleToTransmitAllData_d, senderBuffer)
c
dToC
Checksum
Ack0
Ack1
(reply)
Nak0
Nak1
Seq1
<< Global >> << NetState5 >>


(RDT21) Check AlwaysPossibleToTransmitAllData for 2 but 7 NetState
File Instance Theme Window
Viz Txt Tree Theme Magic Layout Evaluator Next
Projected over 2 sigs
c: 1
dToC: 1
pToD: 1
seq_num: 1
Packet
pToD
seq_num
Seq0
Data
($AlwaysPossibleToTransmitAllData_d, senderBuffer)
c
dToC
Checksum
Ack0
Ack1
Nak0
Nak1
Seq1
<< Global >> << NetState6 >>