RETI DI CALCOLATORI – ***Home Assignment protocolli a finestra scorrevole***

Prima parte

Q1. Indicare –giustificando la risposta– se è possibile o meno che la dimensione della finestra del protocollo Go-Back-N sia ¾ della dimensione dello spazio dei numeri di sequenza utilizzati.

Q2. Un sender Selective Repeat con dimensione della finestra uguale a 4 e con 3 segmenti "in volo" riceve un riscontro non duplicato R. Indicare – giustificando la risposta – se è possibile o meno che tale sender **non** possa inviare nuovi segmenti dopo avere ricevuto R.

Q3. Un sender Selective Repeat con dimensione della finestra uguale a 2N e con N segmenti "in volo" riceve un riscontro non duplicato R. Determinare – giustificando la risposta – quanti nuovi segmenti tale sender potrebbe inviare dopo avere ricevuto R.

Q4. Indicare – giustificando la risposta – quanti nuovi segmenti può inviare un sender Go-Back-N subito dopo avere ricevuto un riscontro R, se subito prima di ricevere R ha N/2 segmenti spediti e non ancora riscontrati, dove N è la dimensione della finestra

Q5. Un sender Selective Repeat con dimensione della finestra uguale ad N e con K segmenti in volo, con $1<K<N$, riceve un riscontro R. Indicare – giustificando la risposta – se è possibile o meno che tale sender, subito dopo aver ricevuto R, possa inviare N nuovi segmenti.

Q6. Sia m la dimensione in bit del campo sequence number utilizzato da Selective Repeat, sia Sf l'indice del segmento più vecchio spedito dal sender e per esso non ancora riscontrato e sia Rn l'indice del prossimo segmento atteso dal receiver. Indicare –giustificando la risposta– l'intervallo dei possibili valori che Sf può assumere rispetto a Rn e a m.

Seconda parte

E1. Consideriamo una variante del protocollo Go-Back-N in cui la prima volta che si verifica un evento che causa la ritrasmissione dei pacchetti presenti nella finestra, questi vengono ritrasmessi tutti. Se l'evento successivo causa ancora la ritrasmissione dei pacchetti, questi non vengono ritrasmessi tutti, ma solo la prima metà. Se di nuovo il primo evento successivo che si verifica causa la ritrasmissione dei pacchetti, vengono ritrasmessi quelli appartenenti al primo quarto dei pacchetti presenti nella finestra. Si procede in questo modo, dimezzando sempre il numero di pacchetti da ritrasmettere (ma ritrasmettendone sempre almeno uno), finché non si riceve un riscontro positivo. In tal caso, ci si riconduce alla situazione iniziale in cui al prossimo evento che causa ritrasmissione si ritrasmette tutta la finestra, per poi (eventualmente) procedere come sopra specificato. Descrivere, utilizzando un automa a stati finiti che utilizza pseudocodice e non descrizione delle attività a parole, il comportamento del sender della variante del protocollo Go-Back-N sopra descritta.

E2. Una rete utilizza una variante del protocollo GoBackN in cui il receiver non invia un riscontro positivo subito dopo aver ricevuto correttamente il pacchetto atteso, ma invia un riscontro (cumulativo) solamente ogni TR istanti di tempo nel caso in cui abbia ricevuto correttamente almeno un nuovo pacchetto da quando ha inviato l'ultimo riscontro. Se non riceve nessun segmento per TR istanti di tempo, torna ad attendere un tempo uguale a TR. Quando invece riceve un segmento corrotto o con numero di sequenza diverso da quello atteso allora invia subito un riscontro "negativo". Descrivere il comportamento di tale receiver utilizzando un automa a stati finiti che utilizza pseudocodice e non descrizione delle attività a parole.

E3. Modificare l'automa a stati finiti che descrive il comportamento del receiver di Selective Repeat utilizzando un automa a stati finiti che utilizza pseudocodice e non descrizione delle attività a parole, in modo che invii un riscontro solo se riceve correttamente il segmento più vecchio tra quelli attesi. Il riscontro inviato deve inoltre contenere il numero di sequenza dell'ultimo segmento passato al livello superiore.

E4. Consideriamo una variante del protocollo Selective Repeat in cui il sender utilizza un unico timer associato al segmento più vecchio in volo e, in caso di timeout, rispedisce solo tale segmento. Descrivere con utilizzando un automa a stati finiti che utilizza pseudocodice e non descrizione delle attività a parole il comportamento di tale sender.

Q1. Indicare –giustificando la risposta– se è possibile o meno che la dimensione della finestra del protocollo Go-Back-N sia ¾ della dimensione dello spazio dei numeri di sequenza utilizzati.

Si. dato che in GBN è sufficiente che la dimensione della finestra sia minore della dimensione dello spazio dei numeri di sequenza utilizzati. Nel caso pessimo infatti il mittente considera in volo i segmenti [base,base+N-1] mentre il ricevente li ha già correttamente ricevuti e attende di ricevere il segmento base+N.

Q2. Un sender Selective Repeat con dimensione della finestra uguale a 4 e con 3 segmenti "in volo" riceve un riscontro non duplicato R. Indicare – giustificando la risposta – se è possibile o meno che tale sender non possa inviare nuovi segmenti dopo avere ricevuto R.

Si. Considerando le quattro possibili situazioni (illustrate sotto) in cui può trovarsi il buffer di spedizione quando il sender riceve R, ciò accade nei primi tre casi se R.ackNum>sendBase (se R non è un riscontro del segmento in sendBase).

sendBase	*sendBase+1*	*sendBase+2*	*sendBase+3*
in-flight	acked	in-flight	in-flight
in-flight	in-flight	acked	in-flight
in-flight	in-flight	in-flight	acked
in-flight	in-flight	in-flight	unsent

Q3. Un sender Selective Repeat con dimensione della finestra uguale a 2N e con N segmenti "in volo" riceve un riscontro non duplicato R. Determinare – giustificando la risposta – quanti nuovi segmenti tale sender potrebbe inviare dopo avere ricevuto R.

Dopo avere ricevuto R il sender potrebbe inviare da 0 a N+1 nuovi segmenti, a seconda di quale segmento viene riscontrato da R e di come sono disposti nel buffer i segmenti in volo. Nel caso pessimo infatti il sender non potrà inviare alcun nuovo segmento – per esempio se il segmento meno vecchio in volo occupa l'ultima posizione della finestra e R non riscontra il segmento più vecchio in volo. Nel caso ottimo invece il sender potrà inviare N+1 nuovi segmenti – per esempio se i segmenti in volo occupano le prime N posizioni della finestra, le successive N posizioni possono essere utilizzate per spedire nuovi segmenti e R riscontra il segmento più vecchio in volo.

Q4. Indicare – giustificando la risposta – quanti nuovi segmenti può inviare un sender Go-Back-N subito dopo avere ricevuto un riscontro R, se subito prima di ricevere R ha N/2 segmenti spediti e non ancora riscontrati, dove N è la dimensione della finestra

Subito dopo aver ricevuto R il sender potrà inviare N/2+X nuovi segmenti, dove X in [0,N/2] è il numero (X=R.ackNum-base se R.ackNum≥base, X= R.ackNum+buffer.size()-base altrimenti: (caso di uso di buffer circolare)) di segmenti in volo riscontrati da R.

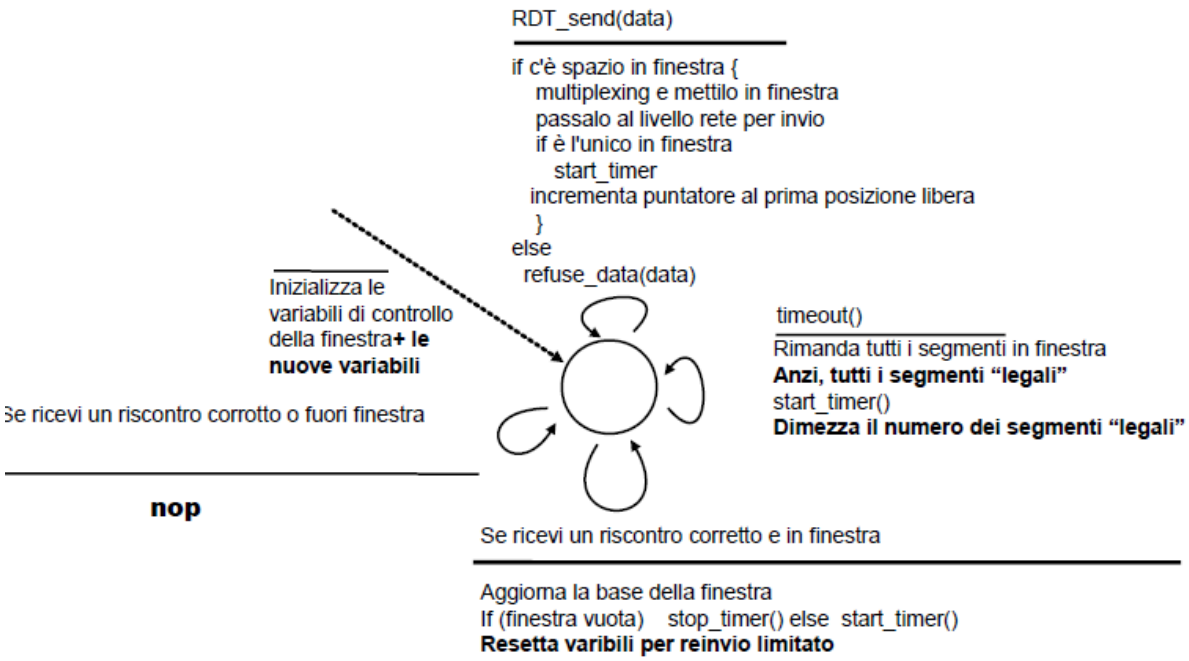
Q5. Un sender Selective Repeat con dimensione della finestra uguale ad N e con K segmenti in volo, con 1<K<N, riceve un riscontro R. Indicare – giustificando la risposta – se è possibile o meno che tale sender, subito dopo aver ricevuto R, possa inviare N nuovi segmenti.

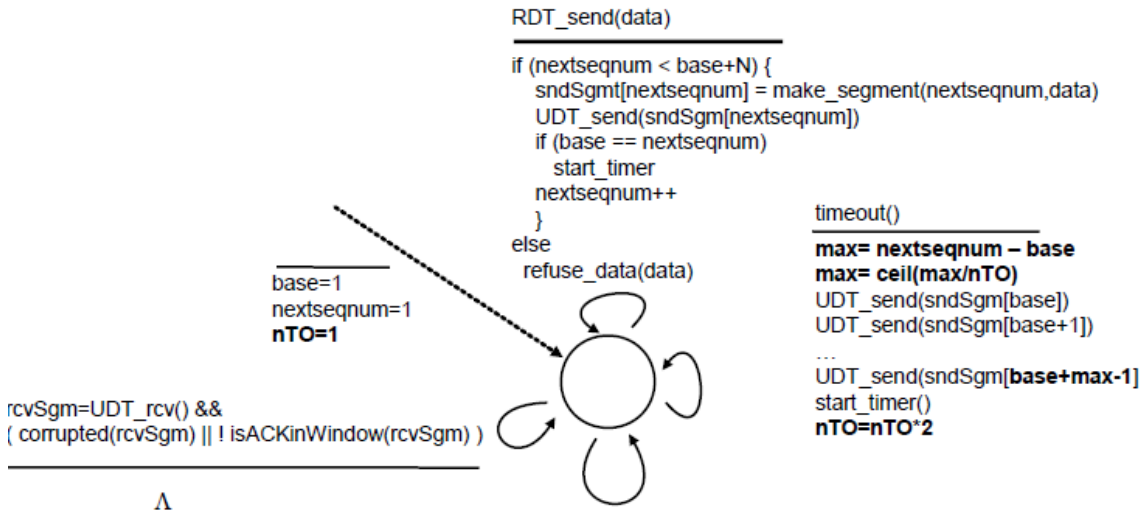
No, non è possibile perché dopo aver ricevuto R il sender avrà ancora almeno 1 segmento in volo (aveva 2 segmenti in volo prima di ricevere R e in Selective Repeat i riscontri non sono cumulativi).

Q6. Sia m la dimensione in bit del campo sequence number utilizzato da Selective Repeat, sia Sf l'indice del segmento più vecchio spedito dal sender e per esso non ancora riscontrato e sia Rn l'indice del prossimo segmento atteso dal receiver. Indicare –giustificando la risposta– l'intervallo dei possibili valori che Sf può assumere rispetto a Rn e a m.

Poiché in SR la dimensione massima della finestra di invio è 2^{m-1}=N/2, abbiamo che:
- se Rn >= 2^{m-1} allora Sf in [Rn-2^{m-1} , Rn] e
- se Rn < 2^{m-1} allora Sf in [0, Rn] o Sf in [2^{m-1}, Rn+2^{m-1}] (caso di uso di buffer circolare)

E1. Consideriamo una variante del protocollo Go-Back-N in cui la prima volta che si verifica un evento che causa la ritrasmissione dei pacchetti presenti nella finestra, questi vengono ritrasmessi tutti. Se l'evento successivo causa ancora la ritrasmissione dei pacchetti, questi non vengono ritrasmessi tutti, ma solo la prima metà. Se di nuovo il primo evento successivo che si verifica causa la ritrasmissione dei pacchetti, vengono ritrasmessi quelli appartenenti al primo quarto dei pacchetti presenti nella finestra. Si procede in questo modo, dimezzando sempre il numero di pacchetti da ritrasmettere (ma ritrasmettendone sempre almeno uno), finché non si riceve un riscontro positivo. In tal caso, ci si riconduce alla situazione iniziale in cui al prossimo evento che causa ritrasmissione si ritrasmette tutta la finestra, per poi (eventualmente) procedere come sopra specificato. Descrivere, utilizzando un automa a stati finiti, il comportamento del sender della variante del protocollo Go-Back-N sopra descritta.

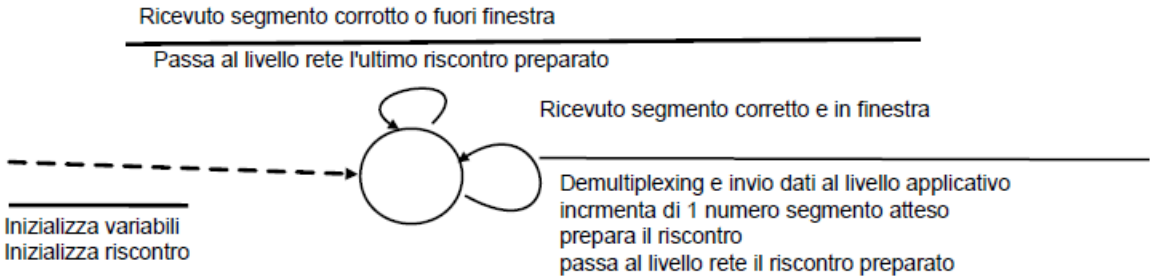


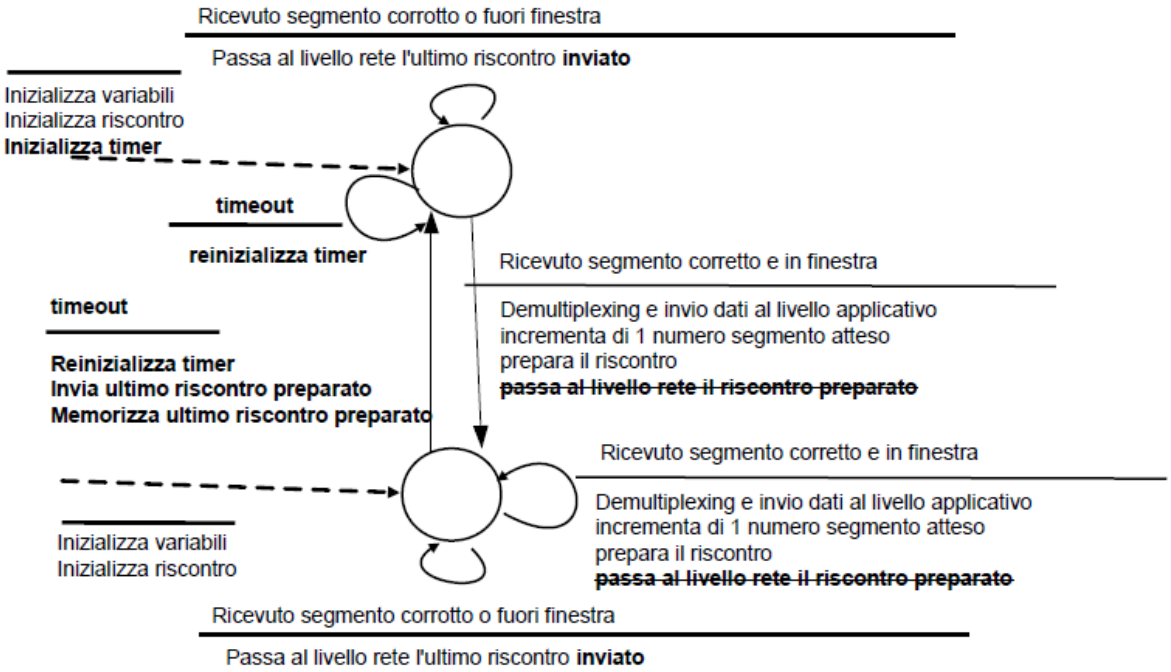


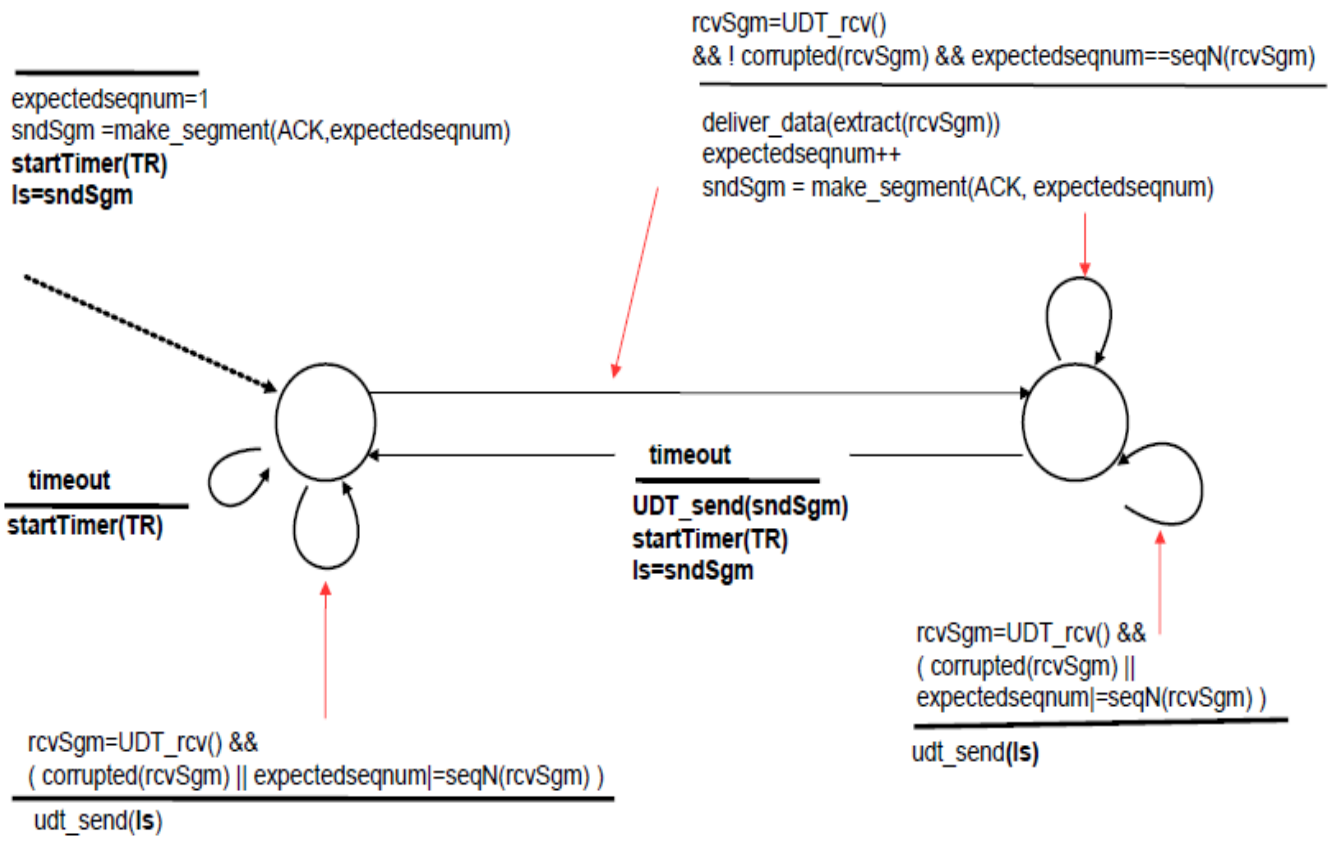
rcvSgm=UDT_rcv() && ! corrupted(rcvSgm) && isACKinWindow(rcvSgm)

nTO=1
base = getacknum(rcvSgm)
If (base == nextseqnum) stop_timer() else start_timer()

E2. Una rete utilizza una variante del protocollo GoBackN in cui il receiver non invia un riscontro positivo subito dopo aver ricevuto correttamente il pacchetto atteso, ma invia un riscontro (cumulativo) solamente ogni TR istanti di tempo nel caso in cui abbia ricevuto correttamente almeno un nuovo pacchetto da quando ha inviato l'ultimo riscontro. Se non riceve nessun segmento per TR istanti di tempo, torna ad attendere un tempo uguale a TR. Quando invece riceve un segmento corrotto o con numero di sequenza diverso da quello atteso allora invia subito un riscontro "negativo". Descrivere il comportamento di tale receiver utilizzando un automa a stati finiti.

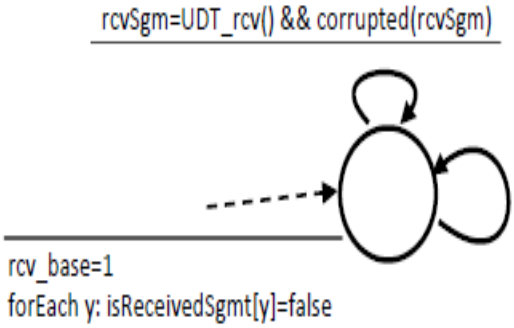





expectedseqnum=1
sndSgm =make_segment(ACK,expectedseqnum)
startTimer(TR)
ls=sndSgm
rcvSgm=UDT_rcv()
&& ! corrupted(rcvSgm) && expectedseqnum==seqN(rcvSgm)
deliver_data(extract(rcvSgm))
expectedseqnum++
sndSgm = make_segment(ACK, expectedseqnum)
timeout
startTimer(TR)
timeout
UDT_send(sndSgm)
startTimer(TR)
ls=sndSgm
rcvSgm=UDT_rcv() &&
(corrupted(rcvSgm) || expectedseqnum|=seqN(rcvSgm))
udt_send(ls)
rcvSgm=UDT_rcv() &&
(corrupted(rcvSgm) || expectedseqnum|=seqN(rcvSgm))
udt_send(ls)

E3. Modificare l'automa a stati finiti che descrive il comportamento del receiver di Selective Repeat in modo che invii un riscontro solo se riceve correttamente il segmento più vecchio tra quelli attesi. Il riscontro inviato deve inoltre contenere il numero di sequenza dell'ultimo segmento passato al livello superiore.

E1.




```
rcvSgm=UDT_rcv() && !corrupted(rcvSgm)
y=seqN(rcvSgm)
if (isReceivedSgmt[y]==false)
  {rcvSgmt[y]=extract(rcvSgm); isReceivedSgmt[y]=true}
if (y==rcv_base) {
  while (isReceivedSgm[rcv_base]==true) do {
      deliver_data(rcvSgmt[rcv_base])
      isReceivedSgmt[y]=false
      rcv_base++
  }
 sndSgm = make_segment(ACK, rcv_base-1)
 UDT_send(sndSgm)
}
```

Nelle due condizioni in alto, bisogna anche inserire il controllo che il segmento ricevuto sia nella finestra: le condizioni diventano

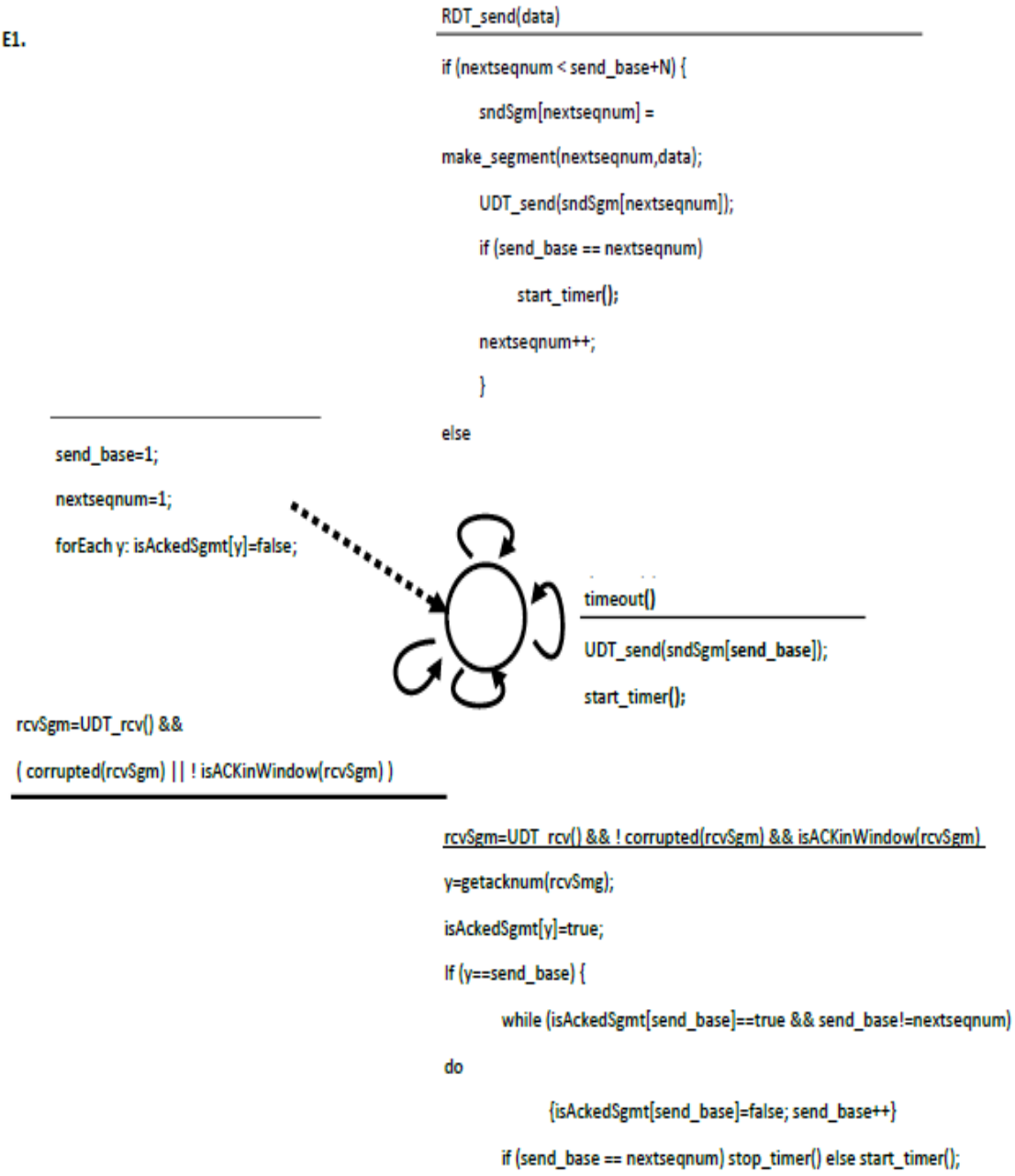
rcvSgm=UDT_rcv() && (corrupted(rcvSgm)|| !isInEWindow(rcvSgm))

e

rcvSgm=UDT_rcv() && !corrupted(rcvSgm)&&isInEWindow(rcvSgm)

E4. Consideriamo una variante del protocollo Selective Repeat in cui il sender utilizza un unico timer associato al segmento più vecchio in volo e, in caso di timeout, rispedisce solo tale segmento. Descrivere con un automa a stati finiti il comportamento di tale sender.

E1.



In **RDT_send(data)** va inserita l'istruzione del ramo else, che è: *refuse_data(data)*