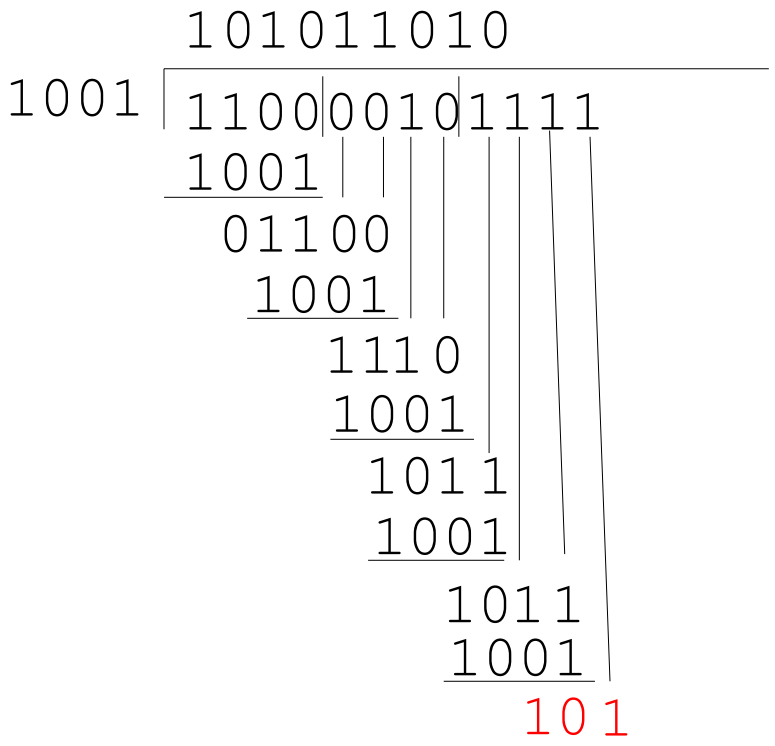
Contents

- How can we provide a reliable service in the transport layer

- Hypotheses
  1. The application sends small SDUs
  2. The network layer provides a perfect service
     1. Packets corruption are possible
     2. No packet is lost
     3. There is no packet reordering
     4. There are no duplications of packets
  3. Data transmission is unidirectional

Example

Assume a segment of 12 bits : 1100|0010|1111
And a divisor of 4 bits : 1001



Cyclical Redundancy Check (CRC)

- Motivation & Principle
  - Improve the computation speed by using polynomial codes instead of positive integers for the division.
- Solution
  - Each number of the division is a bitstream whose coefficients bits are coefficient of a polynome.
    - Ex : 1001 represent $1x^3+0x^2+0x^1+1x^0$ which is x^3+1
  - Sender and receiver do their arithmetic on polynomes modulo 2 (addition of coefficients value is done modulo 2) rather than positive integers (i.e remainder is a polynome).
  - Very nice property of this arithmetic : isomorphic with binary arithmetic without carry : addition and subtraction are equivalent, it's just a XOR.

https://en.wikipedia.org/wiki/Mathematics_of_cyclic_redundancy_checks

Cyclical Redundancy Check (CRC)

- Details
  - Sender and receiver agree on r+1 bits pattern called Generator (G)
  - Sender adds r bits of CRC to a d bits data segment such that the d+r bits pattern is exactly divisible by G (i.e the remainder is zero).

  - All computations are done by using XOR
    - 1011 + 0101 = 1110
    - 1011 - 0101 = 1110

    1001 + 1101 = 0100
    1001 - 1101 = 0100

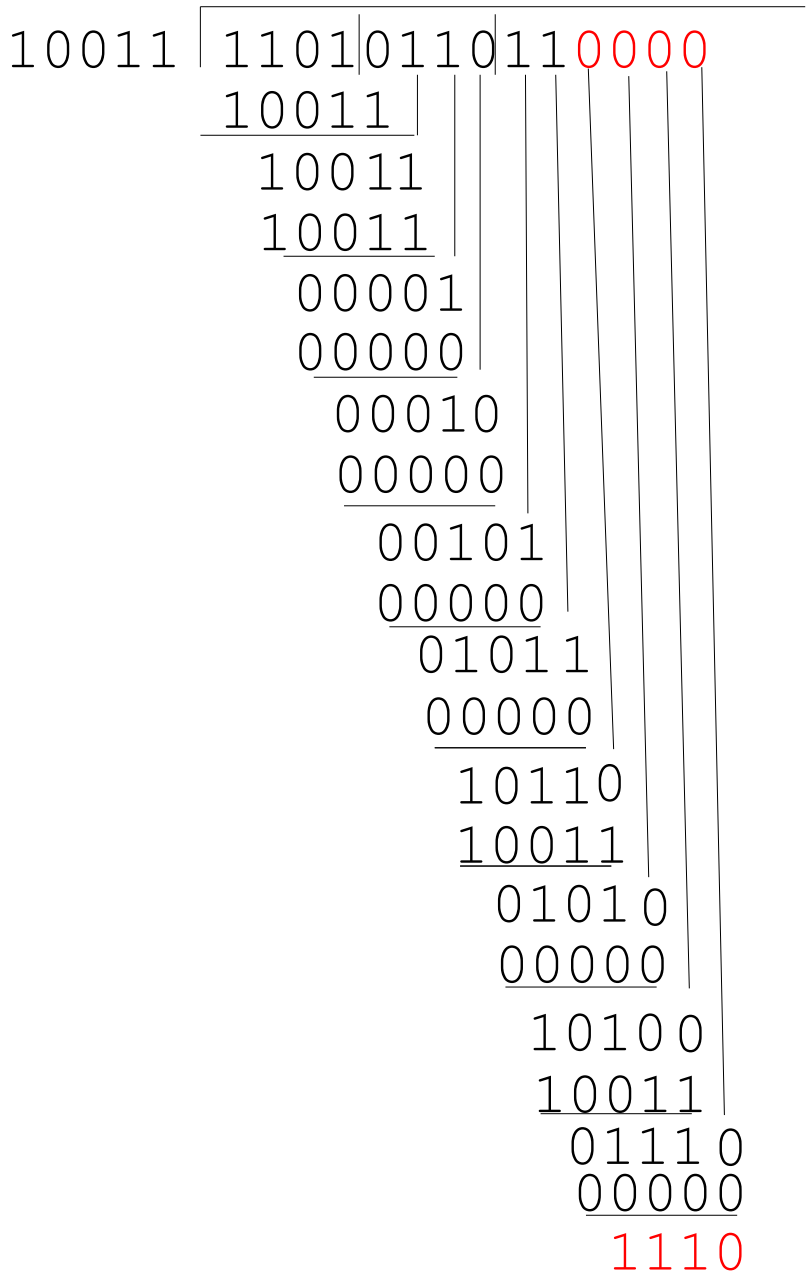
Example

Assume G, 5 bits

10011

on a segment of 10 bits:

1101|0110|11



The message will be :

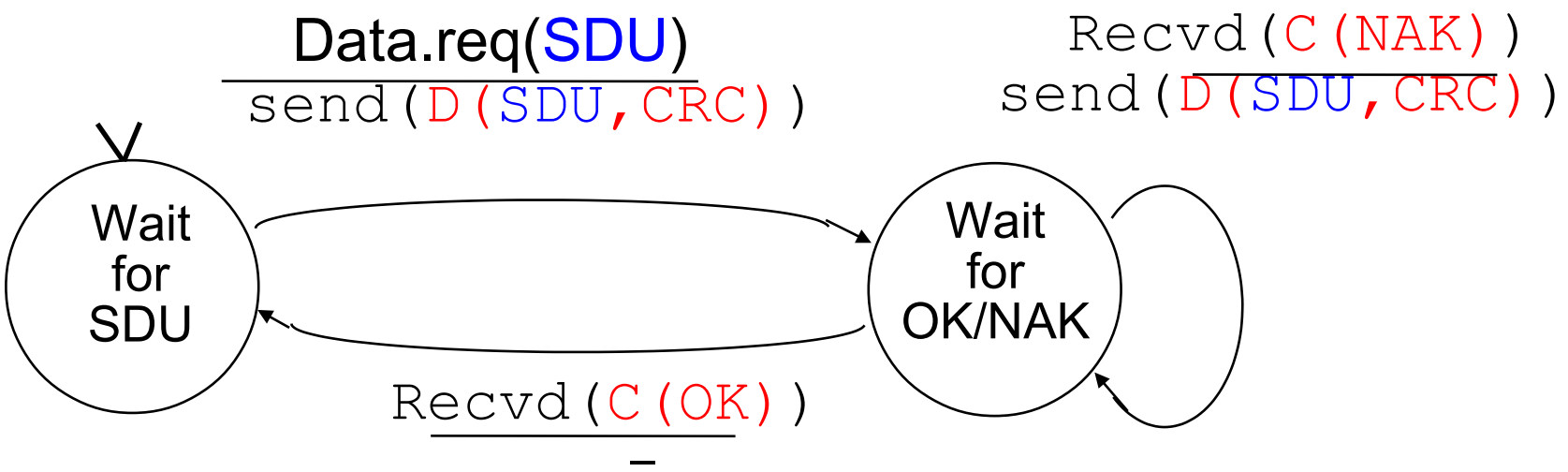
1101|0110|111110

Detection of transmission errors (2)

- Behaviour of the receiver
  - If the checksum is correct
    - Send an OK control segment to the sender to
      - confirm the reception of the data segment
      - allow the sender to send the next segment
  - If the checksum is incorrect
    - The content of the segment is corrupted and must be discarded
    - Send a special control segment (NAK) to the sender to ask it to retransmit the corrupted data segment

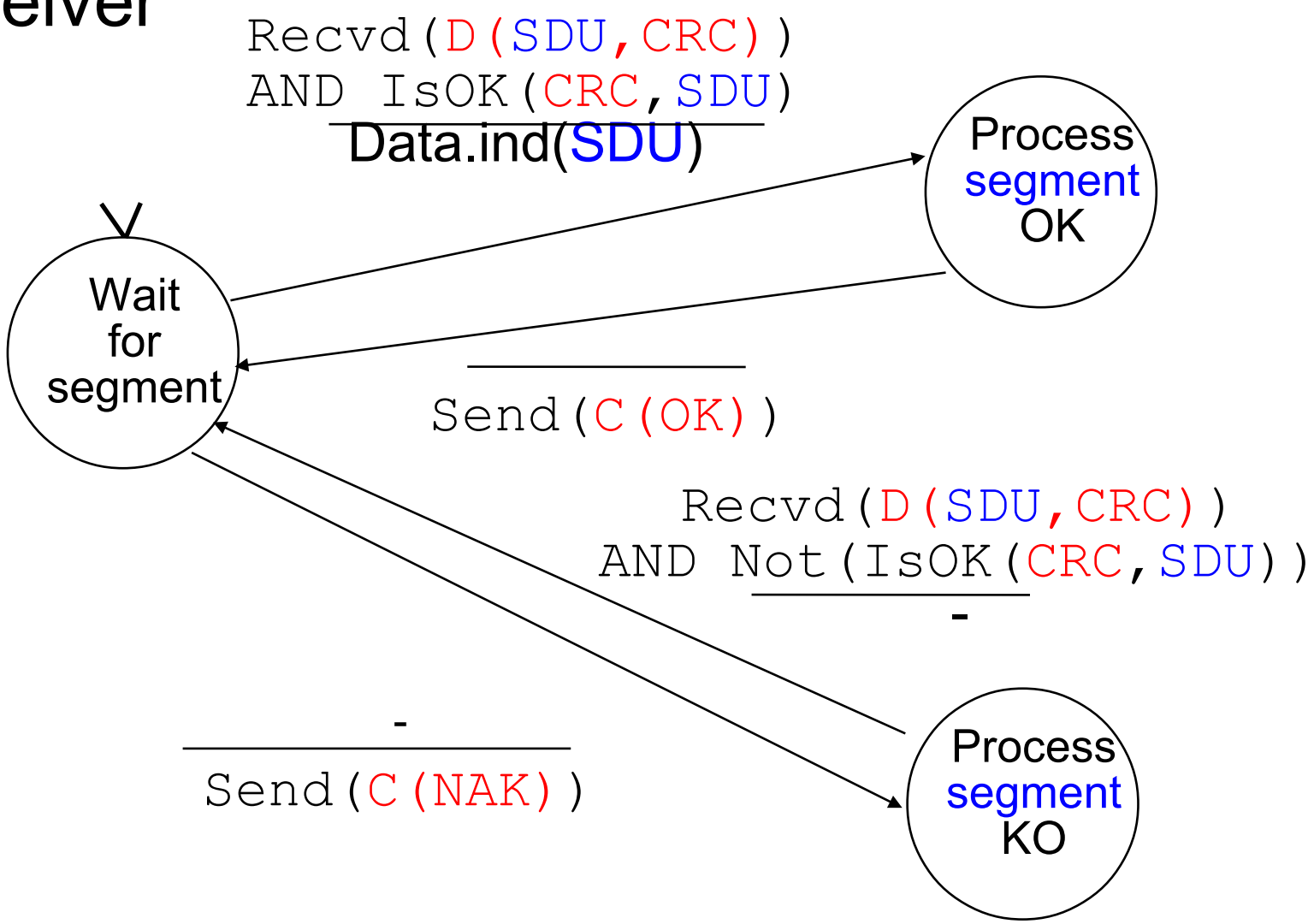
Protocol 3a : Sender

- Sender

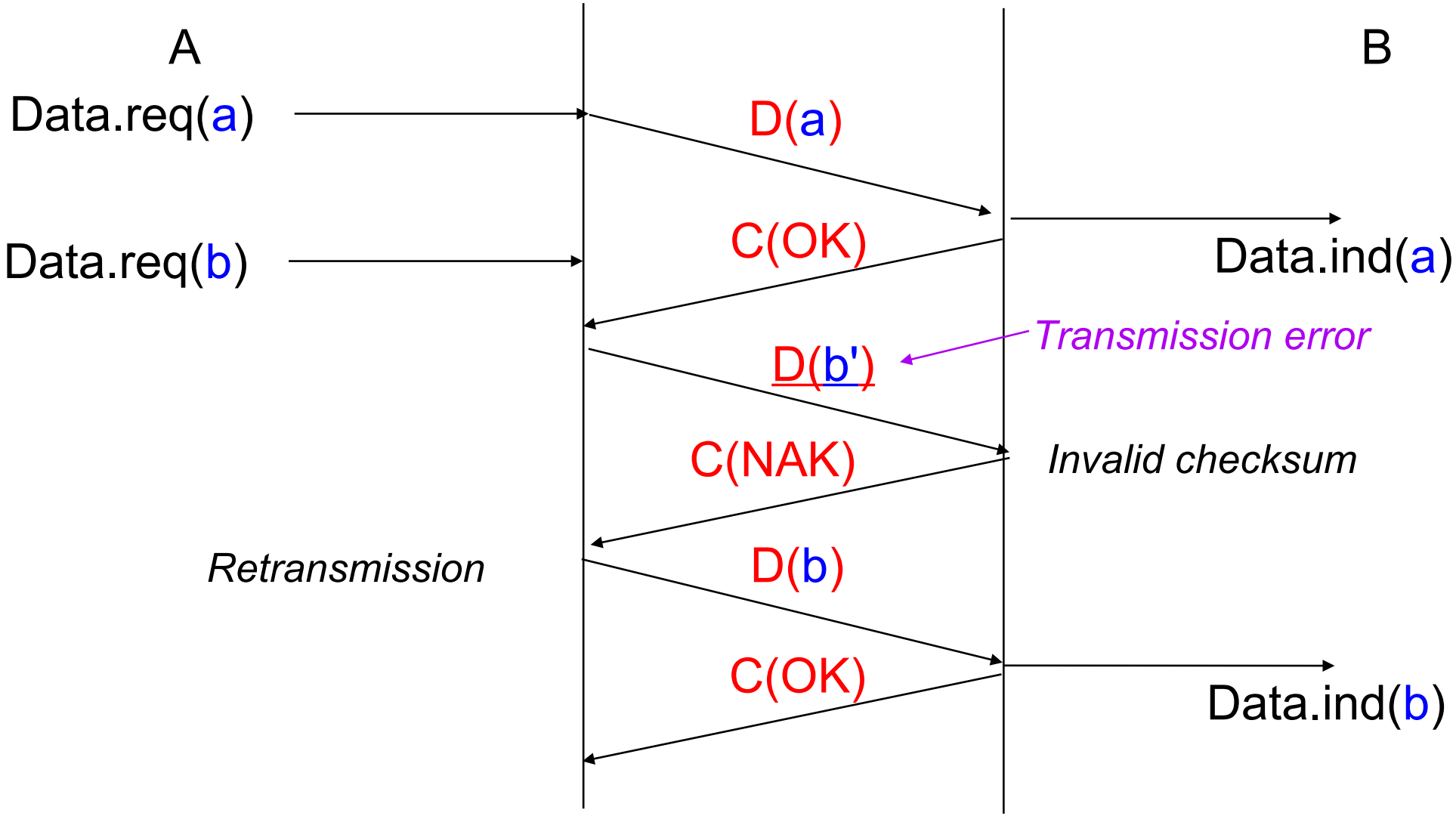


Protocol 3a : Receiver

- Receiver



Protocol 3a : Example

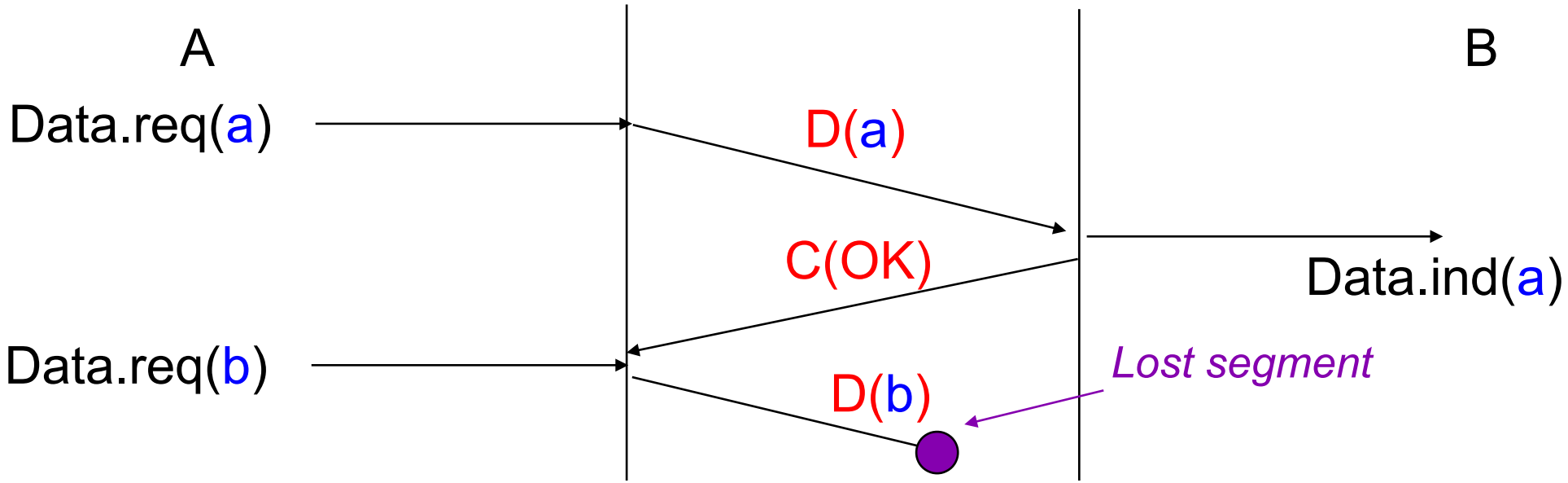


Protocol 3b

- How can we provide a reliable service in the transport layer ?
  - Hypotheses
    - The application sends small SDUs
    - The network layer provides a perfect service
      - Packets corruption are possible
      - Packets can be lost
      - There is no packet reordering
      - There are no duplications of packets
    - Data transmission is unidirectional
  - How to deal with these problems ?

Protocol 3a and segment losses

- How do segment losses affect protocol 3a ?



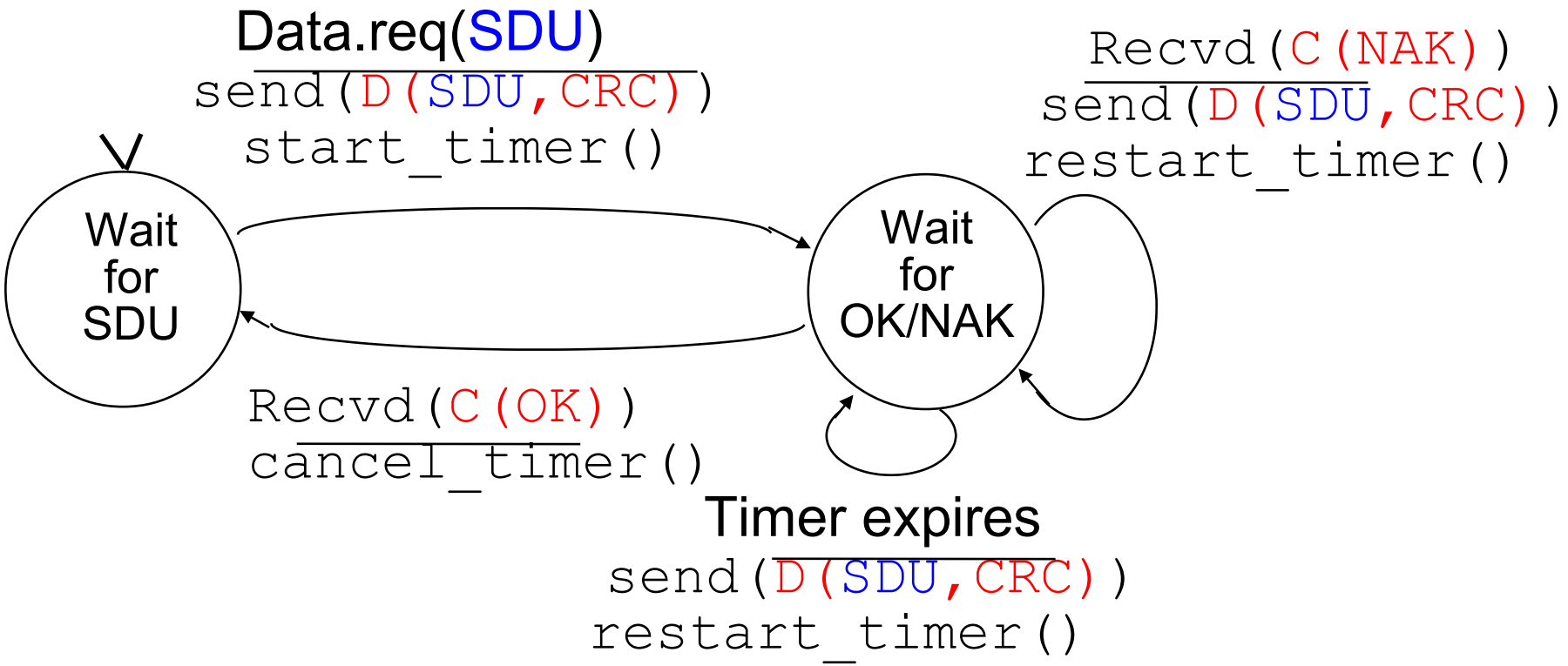
DEADLOCK

A is waiting for a control segment

B is waiting for a data segment

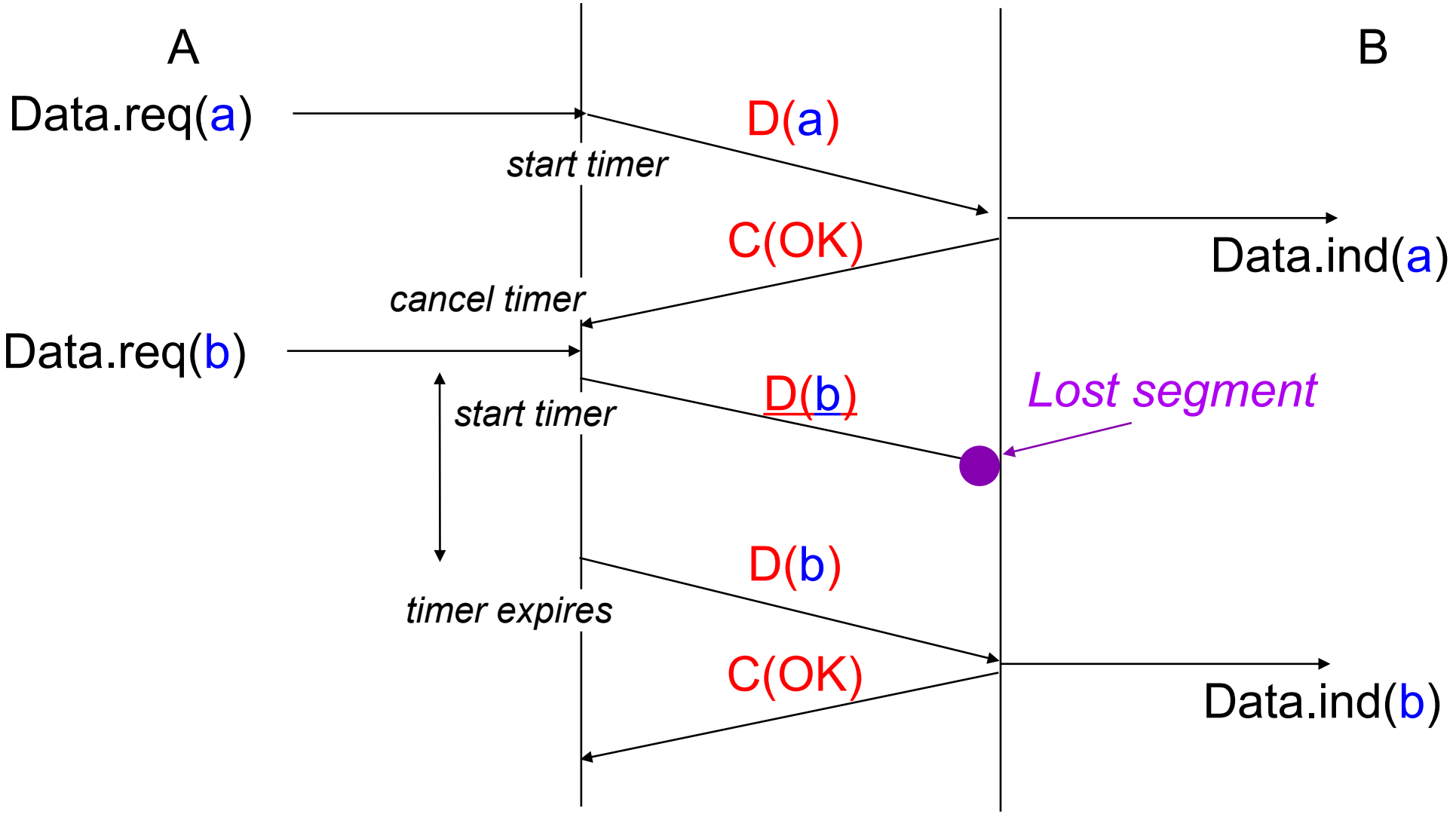
Protocol 3b

- Modification to the sender
  - Add a retransmission timer to retransmit the lost segment after some time



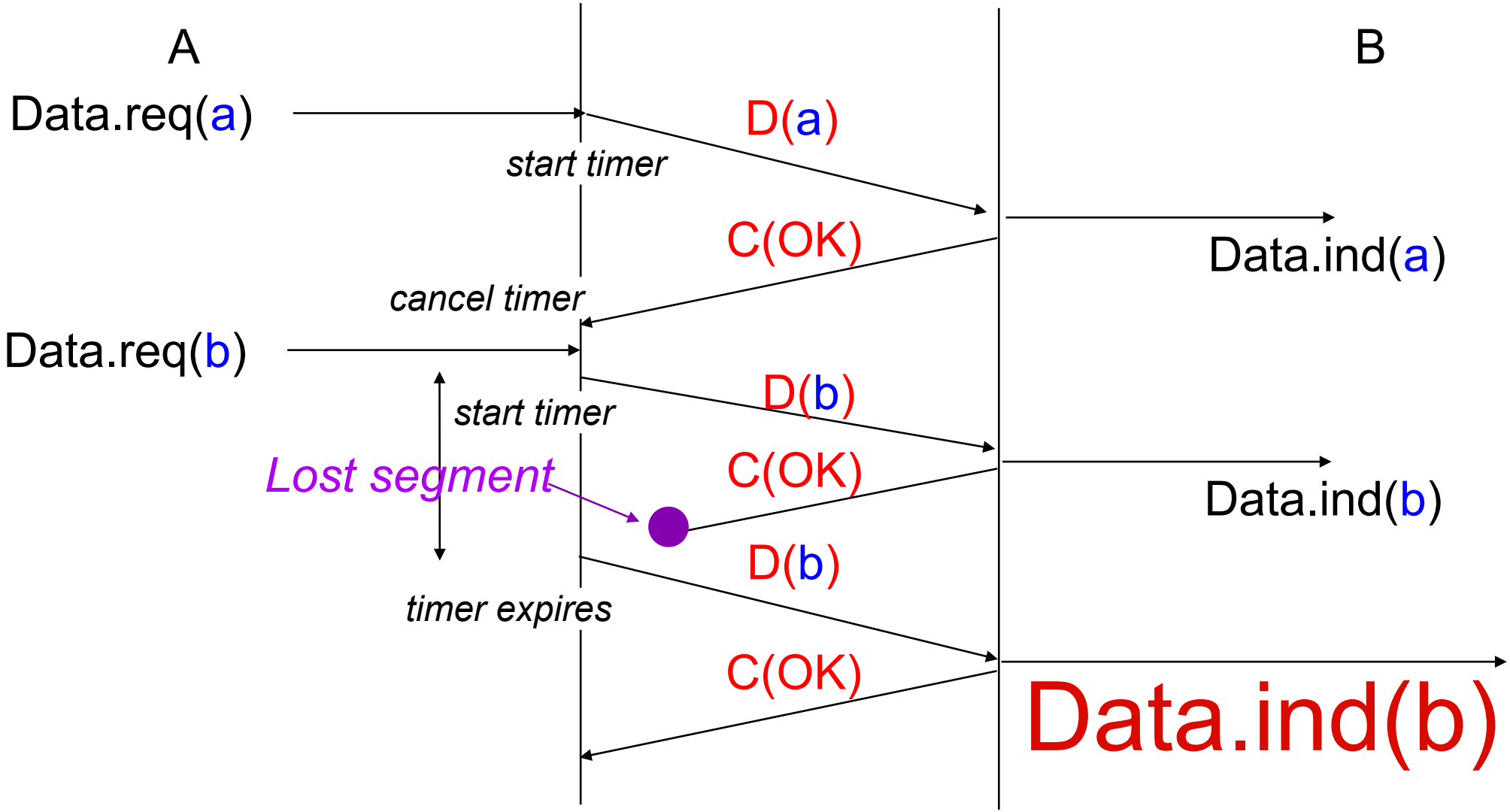
  - No modification to the receiver

Protocol 3b : Example



- Does this protocol always work ?

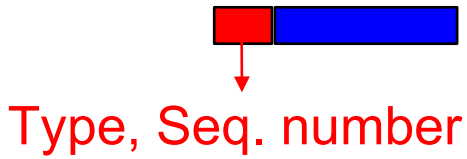
Protocol 3b : Example



- How to solve this problem ?

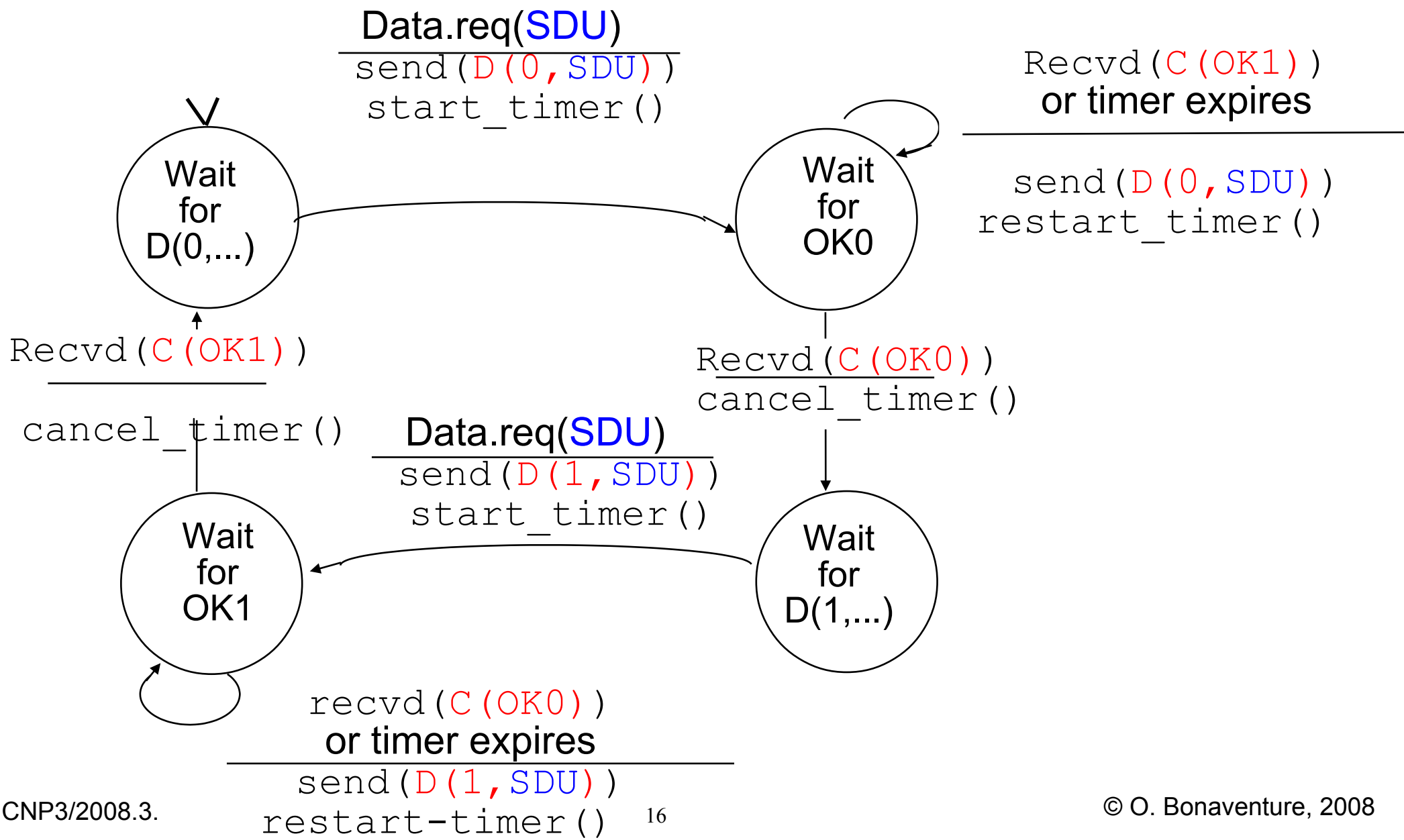
Alternating bit protocol

- Principles of the solution
  - Add sequence numbers to each data segment sent by sender
  - By looking at the sequence number, the receiver can check whether it has already received this segment
- Contents of each segment
  - Data segments
  - Control segments

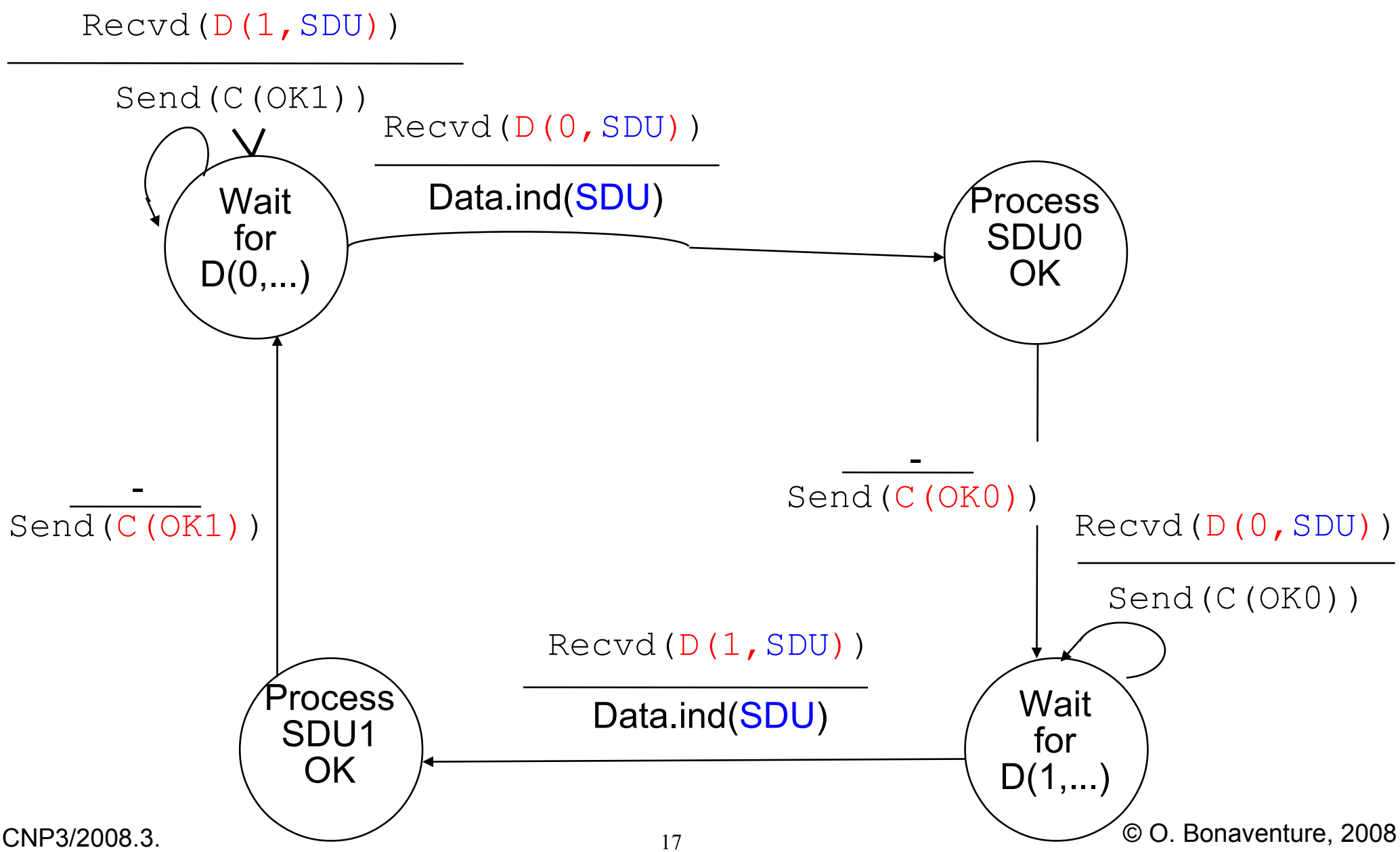


  - How many bits do we need for the sequence number?
    - a single bit is enough

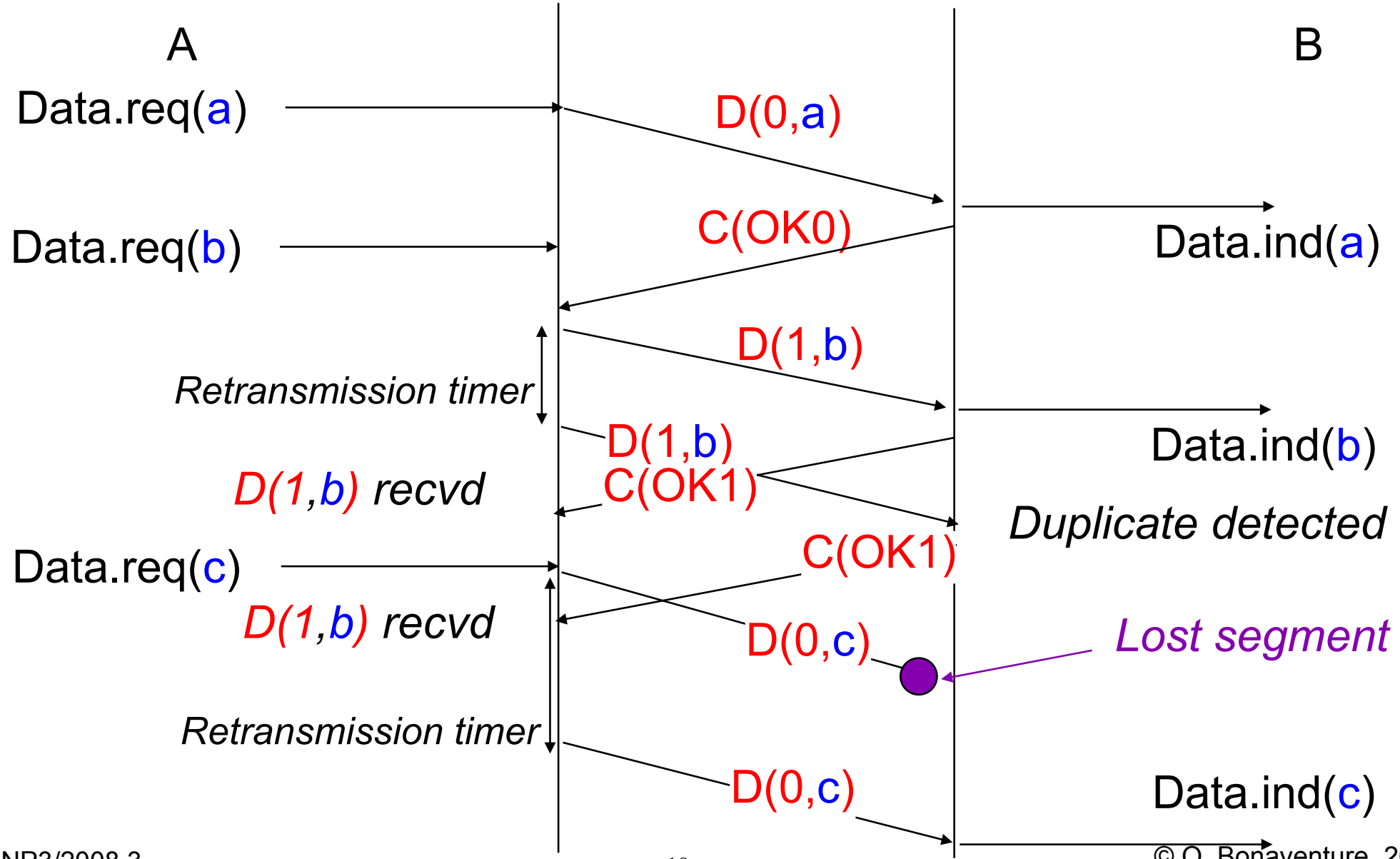
Alternating bit protocol : Sender



Alternating bit protocol : Receiver



Alternating bit protocol : Example



Alternating bit protocol – corrupt messages

- Practice : modify the alternating bit protocol to handle transmissions errors.
  - Contents of each segment.
  - FSM of the sender and the receiver.
- Draw a time sequence diagram of your FSM execution with one transmission error.