NOTES ON THE X.25 PROCEDURES FOR VIRTUAL CALL ESTABLISHMENT AND CLEARING

Gregor V. Bochmann
Département d'Informatique et de Recherche Opérationnelle
Université de Montréal
C.P. 6128, Montreal, Que.
Canada

1. Introduction

The packet level of the CCITT Recommendation X.25 describes how the virtual circuit communication service of public packet-switched data networks can be used. The recommendation contains in particular a specification of how to establish and clear virtual circuits through a given DTE-DCE interface. We comment in this paper on the following three aspects of these procedures for call establishment and clearing:

(a) The recommendation contains state diagrams to clarify the operation of the DTE-DCE interface. The reader may assume that the diagrams show, at each instant in time, the unambiguous state of the interface. This is, however, not always true since the DTE and the DCE may be in different states.

(b) We have analysed the operation of the interface using a finite state validation method, and have found that in some rare cases a conflicting situation (i.e. when the DTE and the DCE are in different states) may sustain through the repetitions of some cyclic behavior.

(c) The procedures of X.25 are very symmetrical. However, the packet level call establishment procedure contains a non-symmetrical element for the handling of call collisions. We consider as alternatives some symmetrical procedures for this purpose.

For each of these aspects we try to explain the problem and give some suggestions for its solution.

2. State description

Figure 1, copied from Recommendation X.25, shows the packet level DTE/DCE interface state diagram for establishing and clearing a virtual circuit through a given logical channel. It gives the impression that the DTE and DCE are always in corresponding states. However, it is to be noted that the two way simultaneous data link which is used for the exchange of packets between the DTE and DCE, the link access procedure (level 2 of X.25), admits the possibility that several packets are in transit between the DTE and DCE in both directions. Therefore the DTE and/or the DCE might do certain state transitions by sending packets before the other side does the corresponding transitions when receiving the packets. This may lead to conflicting situations as explained by Belsnes and Lynning [1]. For

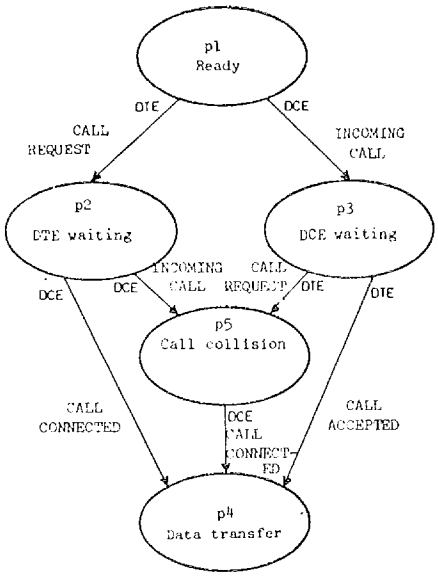

Figure 1a - Call set-up phase

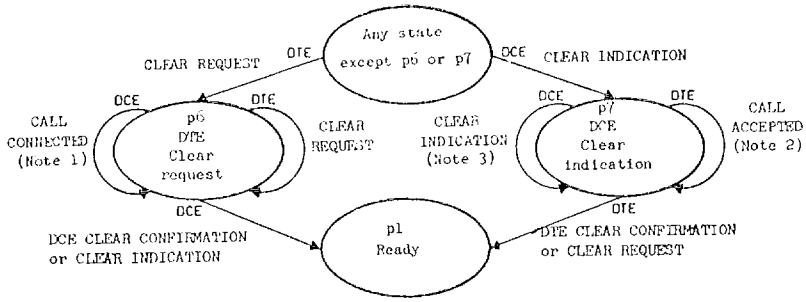

Figure 1b - Call clearing phase

Note 1 : This transition is possible only if the previous state was DTE WAITING (p2).
Note 2 : This transition is possible only if the previous state was DCE WAITING (p3).
Note 3 : This transition will take place after a time out in the network.

Fig. 1: Packet level DTE/DCE interface state diagram for a logical channel (Recommendation X.25)

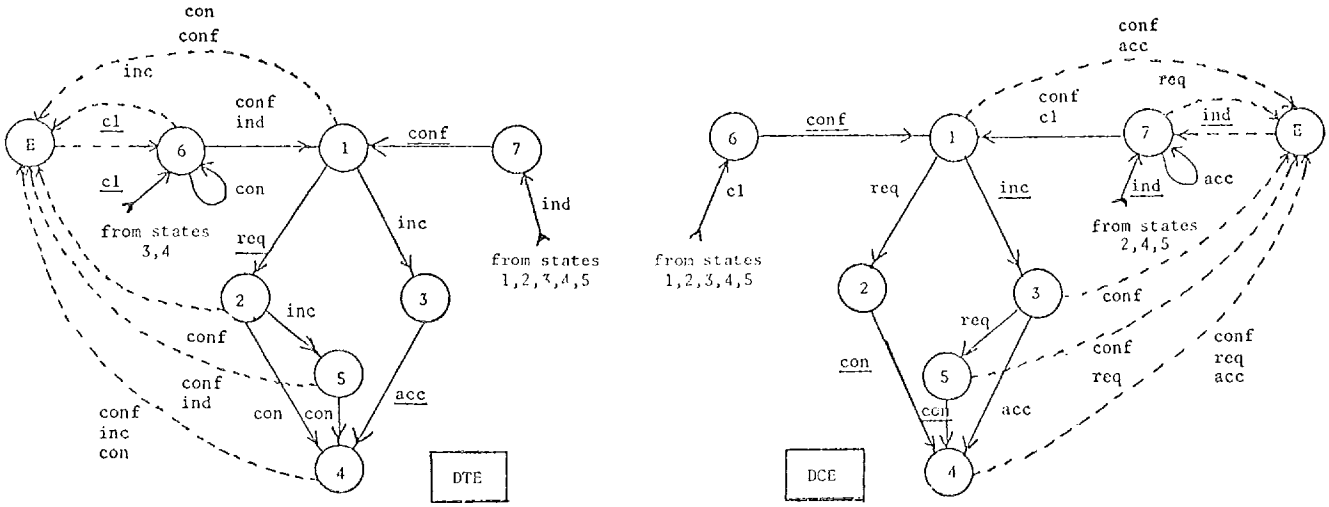

Fig. 2: State diagrams for the DTE and DCE according to Recommendation X.25 with minor restrictions as explained in the text.

req: call request - inc: incoming call
acc: call accepted - con: call connected
cl: clear request - ind: clear indication
conf: clear confirmation

Sending transitions: underlined;
Receiving transitions: non-underlined;
Pointed transitions are for error handling.

example, due to particular timing relations between different packets, the interface may arrive at a "state" where the DTE is in state 3 (DCE waiting) and the DCE is in state 6 (DTE clear request).

Such conflicting situations could also result from the unreliable operation of the environment in which the protocol is implemented. The following reasons could, for instance, be responsible for an unpredicted behavior:

1 - The transmission medium (link level procedures) do not function properly; packets are received erroneously, or are lost or duplicated.

2 - The protocol is not properly initialized.

3 - One party does not follow the prescribed protocol due to a software or hardware bug.

We believe that the possibility of such "conflicting situations" must be taken into account for the validation of any protocol. It is therefore better not to use a single state diagram for describing the communication interface, but to describe both communicating partners by two distinct (and possibly different) state diagrams. In the case of the X.25 packet level virtual call establishment and clearing procedure, we obtain the diagrams of figure 2 [2]. We note the similarity of these finite state diagrams with the original X.25 specification (see figure 1), however, we show explicitely the error handling and have introduced minor restrictions, as explained in the section below.

3. Stability

In the section above, we pointed out that, for different reasons, though exceptionnally, the DTE and DCE may be in different states. This is what we called a conflicting situation of the interface. We believe that a good protocol should show stability in respect to such situations, in the sense that it should recover directly to its normal mode of operation in the case of an initial or intermittent perturbation in the synchronization of the two communicating subsystems, introduced for whatever reason. We have analysed the X.25 virtual call establishment and clearing procedure using a finite state validation method [2], and found that the procedure is not completely stable (in the sense explained above).

For simplifying the analysis of the protocol, we have made two minor restrictions in respect to the original X.25 specifications:

(a) a *cl* (clear request) and an *ind* (clear indication) may not be sent from the states 1,2,5 and 1,3 respectively, and

(b) the DTE and DCE do not receive messages in the states 3,7,E and 2,5,6,E respectively; incoming messages have to wait until the subsystem is in a receiving state.

Reasons for introducing restriction (a) are given by Belsnes and Lynning [1]. Without these restrictions the procedure is even more unstable.

The results of the analysis are shown in the Table and in figure 3. The table shows for a number of different cases and for each possible state of the DTE, the states in which the DCE could possibly be during a period of reliable operation and at an instant when no packets remain within the queues of the link level access procedure. We note that cases (d) and (e) only apply when the link access procedure does not function correctly. The results for case (b) show that certain conflicting situations, once introduced by one of the mechanisms discussed in Section 2, may remain during a period of reliable operation. The reason for this is the possibility of undesired cycles within the operation of the protocol not leading back into the normal mode of operation. The cycles are shown in figure 3. For example, cycle (2) starts in a situation where the DTE is in state 3 and the DCE is in state 6 and no packets are in transit. The cyclic transition occurs when the DTE sends a *cl* (clear indication) and simultaneously (i.e. while the packets sent are still within the queues of the link access procedure) the DCE sends successively a *conf* (clear confirmation) and *inc* (incoming call), which is allowed according to the procedures. The reception of these packets will lead back to the situation in which the cycle started.

These undesired cycles could be avoided by introducing in state 1 of each subsystem (DTE and DCE) a time delay larger than the transmission time of the link access procedure. If these time delays are included in the protocol then the protocol is stable in respect to any perturbations due to an unreliable environment, as indicated by the results in the table. More details are given in [2].

Fig. 3: Undesired cycles of operation for the protocol of figure 2.

Table: Adjoint states for the X.25 virtual call connection protocol

For a number of different cases the table contains, for each state of the DTE, the states in which the DCE could possibly be during a period of reliable operation at an instant when no packets remain to be delivered by the link level procedure.

- basic protocol (see figure 2), and error-free transmission medium:

  case (a): with initial synchronization

  case (b): without initial synchronization (the DTE and DCE are initially in conflicting states)

- same protocol with time delay in state 1, (see text) with or without initial synchronization

  case (c): error-free transmission medium

  case (d): with detected transmission errors

  case (e): with detected transmission errors and packet loss

state of DTE / case	1	2	3	4	5	6	7	E
(a)	1	2	3	4	5	6	7	-
(b)	1	2	3,6	4	5	6	7,2	-
(c)	1	2	3	4	5	6	7	-
(d)	1,E	2,E	3	4,E	5,E	6,E	7	1,3, 4,7
(e)	1,3,7,E	1,2,3,4, 6,7,E	3	3,4,7,E	3,4,5, 7,E	1,3,4, 6,7,E	7	1,3, 4,7

4. Symmetry

Although, in the application between a public data network and a user DTE, the communication interface does not need to be symmetrical, the X.25 protocol is symmetrical in most of its aspects. As far as the establishment and clearing of virtual calls is concerned, the finite state machines for the DTE and DCE in figure 2 are equivalent except for the actions taken in state 5. (We note that the packets *req* and *inc*, *acc* and *con*, and *cl* and *ind* respectively are identically coded, so that the DTE states 2,3,6 and 7 correspond to the DCE states 3,2,7 and 6 respectively). The difference in state 5 is the handling of call collisions: the DTE has priority over the DCE; the latter will normally confirm the call requested by the DTE.

For certain applications, such as the direct communication of two DTEs or internetwork communications, a balanced mode of operation with X.25 would be desirable. For this purpose the problem of call collision could be handled in one of the following ways (each one being a refinement of the X.25 specifications):

1 - In state 5, both subsystems send a clear packet (*cl* or *ind* respectively).

2 - Both subsystems can play the role of either party, DTE or DCE .

The second alternative presents the disadvantage that the role of each subsystem must be chosen prior to the operation of the protocol. This could be done through the operation of a human operator or by a lower level protocol. The first alternative presents the problem that in the case of calls colliding on the same logical channel number, a racing condition occurs. In order to cut short the racing, one could choose different fixed time-outs for the two subsystems. This choice seems to us just as problematic as the choice necessary in the second alternative. Another possibility is to choose the time-outs randomly.

We also note that, if alternative 1 is adopted, call collisions could occur on all virtual channel numbers, whereas when alternative 2 is adopted it would make sense to extend the specifications of X.25 indicating that the next free channel number is to be chosen differently by the DTE and the DCE (for instance in increasing and decreasing order, respectively). Then call collisions can only occur on the last free channel number. (We note that in addition to these call collisions on a given logical channel number at the DTE-DCE interface, end-to-end DTE-DTE call collisions may occur which are not detected by the X.25 procedures, and must be handled by higher level protocols).

5. Conclusions

The X.25 virtual call establishment and clearing procedure can be described by the two finite state machines of figure 2, which represent the operations of the DTE and DCE respectively. We have noted that the use of two distinct state diagrams describing the two communicating partners provides a clearer specification of the protocol than the use of a single diagram showing the "state" of the interface.

A validation analysis [2] shows that the protocol is not completely stable in respect to perturbations due to special circumstances [1] or an unreliable environment. Some undesired cycles of non-synchronized operation may pertain for some time after the occurrence of an initial or intermittent perturbance in the synchronization between the DTE and DCE. Although it is very improbable that such cycles will occur during the operation of the protocol, a good protocol design should avoid such instabilities. In this case, the cycles can be avoided by introducing appropriate time delays.

For a balanced mode of operation, two alternative methods are discussed for the handling of call collisions on a virtual channel number. The methods either involve the choice of different priorities for the two subsystems, or admit the possibility of racing. We wonder whether there exists any (other) method that is completely symmetrical in respect to both subsystems and avoids racing conditions completely?

Acknowledgement:

We thank D.F. Weir for pointing out an error in an earlier version of this paper. Some financial support from the National Research Council of Canada and the Ministère de l'Education du Québec is gratefully acknowledged.

References

1. D. Belsnes and L. Lynning, "Some problems with the X.25 packet level protocol", INWG Protocol Note 55 (1976), see also Computer Communication Review, this issue.

2. G.V. Bochmann, "Finite state description of communication protocols", Publication # 236, Département d'informatique et de recherche opérationnelle, Université de Montréal, July 1976.