SDL Specification and Verification of Connection Establishment and Release Protocol

S.KAMPIRELLIS, S.TRIANTAFYLLOU and A.ANDREATOS
Engineering Dept.
Hellenic Air Force Academy
Dekeleia Air Force Base
Attica, TGA-1010
GREECE

Abstract: - In this paper a connection establishment and release mechanism of a full duplex communication protocol is specified. The specification is written in SDL graphical [GR] representation. The protocol phases described here are destined for *unreliable* and *noisy* channels. In order to cope with such an environment, the following measures are taken:

1. Connection establishment is done using *double handshaking* for increased reliability and medium testing purposes.
2. Connection release may also be initiated by a physical layer interrupt such as synchronisation loss or node failure; this suits perfectly to *wireless* channels.

This work is an improvement over previous related specifications of connection establishment and release protocols.

Key-Words: - SDL, FDT, specification, verification, protocol, connection, establishment, release, handshaking, unreliable, wireless, noisy channel, physical medium. CSCC'99 Proceedings, Pages:5421-5425

1 Introduction

1.1 Formal description techniques

Formal Description Techniques (FDTs) are languages and methods used for mathematical description of (software and/or hardware) system development, which ensure mathematical precision and tractability. Conventional descriptions of systems are usually given in natural language or diagrammatic forms; such descriptions are not always clear or unambiguous, frequently contain errors and/or omissions, and are often hard to analyse. FDTs are used for the development and verification of systems which are [1]: complex, concurrent, safety-critical, quality critical, security critical and standardised (to ensure global compatibility and unique interpretation). Thus, FDTs were developed to ensure: 1) unambiguous, clear and concise specifications; 2) completeness of specifications; 3) consistency of specifications; 4) tractability of specifications; and 5) conformance of implementations to specifications [1].

1.2 Brief history of SDL

One of the areas where FDTs have been actively used is (tele)communications [1]. ITU-T (ex CCITT) is strongly interested in FDTs; it has expressed its interest so early that it standardised an early version of **Specification and Description Language (SDL)** in 1976. SDL is a formal language for the specification of interactive distributed systems. It provides two types of notation: graphical (GR) and textual (TX) [2]. Although SDL was initially intended to protocol specification of telecommunications applications, its use is now being extended to other areas. Through successive four-year study periods, SDL has evolved from an informal diagrammatic notation to a fully-fledged FDT and has superseded its competitors, ESTELLE and LOTOS.

SDL is based on an extended finite state machine model, supplemented by features for specifying Abstract Data Types (ADTs), this combination being supported by a complete formal semantics. SDL provides constructs for representing structures, behaviours, interfaces and communication links. In addition, it provides constructs for abstraction, module encapsulation and refinement. An SDL-described system is composed of blocks interconnected to each other and to the environment by means of "channels"[1,2]. All these constructs support the representation of a variety of telecom system specifications, including services and protocols.

SDL is broadly used by the telecom community and is well supported by a variety of tools, many of which are commercially available.

1.3 Link Control Protocols

In this paper we investigate the connection establishment and release protocol phases because of their practical use in many real-world applications. Two fundamental versions of the specific protocol exist: a) **single handshaking** and b) **double handshaking** [3]. An SDL specification of the single handshaking protocol can be found in [1]. This paper is therefore organised as follows: section 2 gives a generic description of the protocol; section 3 is an informal specification of the protocol; section 4 contains the SDL protocol formal description and finally, section 5 presents results obtained and draws some conclusions.

2 Generic Description of the Link Control Protocol

A description of the generic properties and the services provided by the data link layer to its users above, is the first and one of the most important steps towards its formal specification. In most reference models, such as ISO/OSI and TCP/IP, the Data Link Layer is the interface between the Physical Layer and the Network Layer. While the former handles the transaction of raw bits, the latter routes the packets of data within the subnet borders. In order for the network layer to provide its services, the data link layer has to deal with the problems posed by an **unreliable physical medium**. These difficulties range from the relatively high Signal-to-Noise ratio and the negligible bit error rate found in fiber optics, to the low Signal-to-Noise ratio and frequent loss of signal during the handoffs found in wireless media (e.g. cellular telephones). Due to the varying difficulties and impairments posed by the transmission medium, many different protocols have been developed during the past decade, depending on the user needs.

To deal with the situation described above, an appropriate protocol should be developed. This protocol should assist the task of establishing and releasing a logical link between two communicating parties (nodes of a network). With such a link established and properly working, the network layer is able to provide the service primitives **Send Packet** and **Receive Packet** to the layer above. These are the default service primitives provided and are independent of the type of service provided by the Network layer. More specifically, regardless whether the latter provides a connection-oriented or connectionless service, the connection establishment is the procedure activated at communication startup. When the data exchange (two-way communication) has finished, the disconnection phase is initiated, which releases the channel allocated to the data transfer.

3 Informal Description of the Connection Establishment and Release Protocol

A link is the only way for two communicating nodes to exchange messages. Since the nodes will be described using the finite-state-machine concept, they should be in the appropriate state for a message transaction in the form of frames (cells in ATM). It is assumed that the nodes start from an initial state, inadequate to exchange messages. In order for them to reach a state where this form of transaction is possible, a special procedure of exchanging messages has to be followed. This procedure is well known as **handshaking**. It consists of some sort of questions posed by the initiator (the node wishing to connect) and the same sort of answers sent by the responder back to the opposite side. The duration of this procedure and its size (measured in messages) are protocol dependent.

According to the protocol, the handshaking can be **single** or **double**. In the first case, the initiator simply sends a frame to the responder informing it about the connection that is to come. The responder then either accepts the request by acknowledging the frame sent, or rejects it by sending a frame back at the initiator informing it of its intentions, or does not answer at all. If the initiator desires a connection strongly, it will try again later, when its timer expires. In the second case, the initiator notifies the responder through a request. The latter may either accept it and send back a positive answer, or reject it and deny the connection. The opposite side has then two choices: the first choice is to accept the response and continue with the connection or disconnection respectively; the second one is to deny the response of the responder and to commence the sequence again. The first alternative results to the successful end of the connection phase (and the beginning of the data exchange phase), or to the beginning of the disconnection phase. The second one repeats the actions described above.

After the designer has determined what kind of handshaking it wishes to implement, a new question arises: which node will be in control of the whole procedure just described? It is this node

which will decide whether to initiate a connection or a disconnection sequence. The developer here is faced with two choices. The first one is to have one control node and the second is to have both of them control the procedure. The choice depends on the system. If the system in inherently bi-directional (as it is in this paper), the protocol is characterised as **balanced**. The other choice is a **master-slave** protocol.

Another important issue is the exact sequence of the signals which has to be followed, in order to consider the connection establishment (or the disconnection) successful. Obviously, this matter cannot be resolved with just a simple reference (such as the one above) to the messages which should be exchanged between the nodes. The only solution is the use of a formal description technique like SDL. This need is a lot easier to understand if the characteristics of the transmission medium are taken into account. More specifically, the medium can lose or corrupt the frames in the channel; in this case reordering is not considered as it is in the data exchange phase, because the node sends a signal only after it has received one. An important detail is that the whole procedure is measured by timers which set the time limits for a frame to be accepted.

Figures 1 and 2 describe usual situations during the connection establishment. Figure 1 shows the connection setup without any problems; Figure 2 shows connection setup with frame loss. It is obvious that these pictures cannot substitute the SDL graphical notation in any way.

4 SDL Formal Description

Part of the SDL formal description is shown below in Figures 3 to 5. The communication system typically consists of two nodes A and B wishing to communicate and a communication channel (medium) as shown in Figure 3. The transmitter (node A) and the receiver (node B) are assumed to use a proper transfer protocol to cope with the unreliable medium (such as the sliding window protocol described in [4]). Therefore the medium is described as a block named Medium containing two processes (Fig. 4). The first process is called MediManager and generates random events which make the whole medium unreliable. The second process is called MessageManager and handles the messages according to the random events generates by MediManager. As a result, the Block Medium may transmit correctly, corrupt or lose incoming messages. This can also model random failures happening in lines or switches of a network falling across the communication path. Each of the nodes becomes aware of the events above by appropriate signals.

Node A is shown in further detail in Figure 5; it basically consists of an entity called EntityA, which is in fact a finite state machine, reacting to the events described above. At system start-up the machine is in a state inappropriate for data exchange. Only by following specific procedures (formally described in [4]), will the machine become ready to communicate and report this successful link establishment to the layer above.

5 Results and Conclusions

The following results and conclusions can be drawn out of this work:

1. Before writing a formal description of a system, it is worth to understand and formally describe its infrastructure first.
2. When describing a system in SDL, its boundary with the environment is completely up to the designer. This matter usually depends on the desired point of view. Furthermore, it is important to clearly define system boundary. In our case block Medium describes the part of the whole system which handles (from the transmitter to the receiver, top-down and vice-versa) fragmentation of the frame to bits, computation of the checksum (CRC), bit or character stuffing (depending on the implementation), medium access (TDMA, CDMA, etc.), transmission via the medium, reception, removal of the stuffed elements, recomputation of the checksum and reassembly of frames.
3. The type of errors a system may cause need to be carefully stated. It is vital for the protocol designer to know exactly what kinds of errors may occur in a medium and how they can be modelled [1]. In our case, a point-to-point link may corrupt or lose messages, or experience other failures.
4. The system described in this paper has the ability to set up a connection in case the medium is very unreliable and failures are frequent, in contrast to the common transmission media used (optical fibres, coaxial cables). Therefore, it is ideal for fault-tolerant military wireless communications, in an extremely loaded environment.
5. The nature of the environment in which the protocol is designed to operate posed many restrictions and requirements during the design phase. The complexity of the protocol was dictated by the requirement to recover from frequent failures (for more information see [4]).

References:

[1] Kenneth J. Turner, Editor, "*Using formal description techniques*", John Wiley & Sons, 1993.

[2] J. Ellsberger, D. Hogrefe and A. Sarma, "*SDL: Formal Object-Oriented Language for Communicating Systems"*, Prentice-Hall, 1997.

[3] Bertsekas & Gallager, "*Data networks"*, 2nd ed., Prentice-Hall, 1992.

[4] S. Kampirellis, S. Triantafyllou & A. Andreatos, "Specification and Verification of a Sliding Window Protocol", to be presented in "Telekomunikace '99" Conference, Brno, CZ, Sept. 1999.

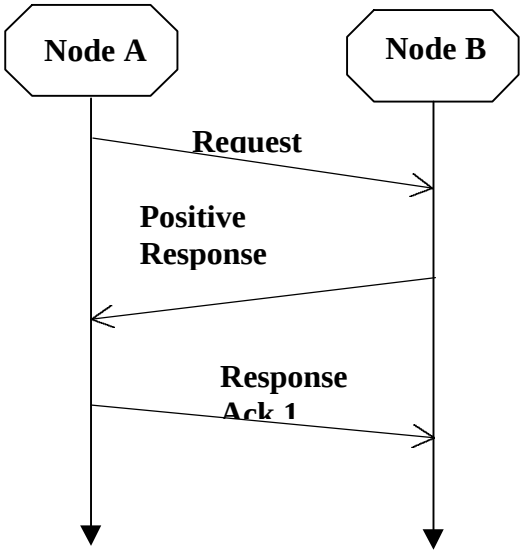

Figure 1
Connection establishment without problems

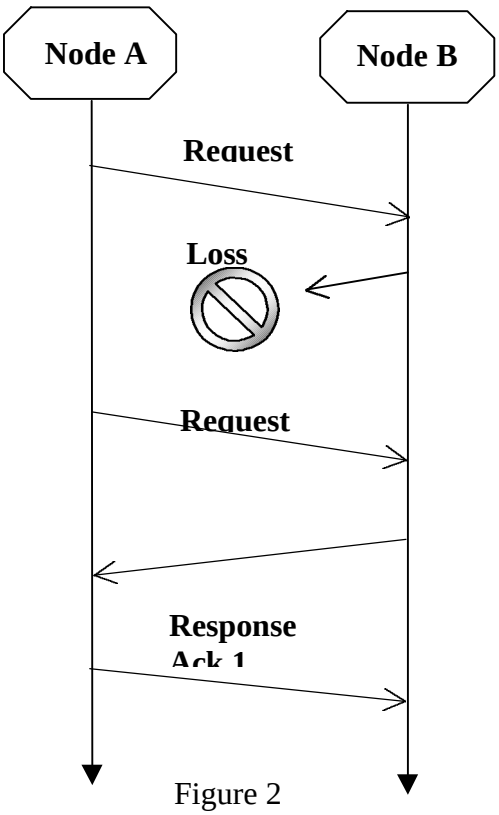

Figure 2
Connection establishment with loss problem

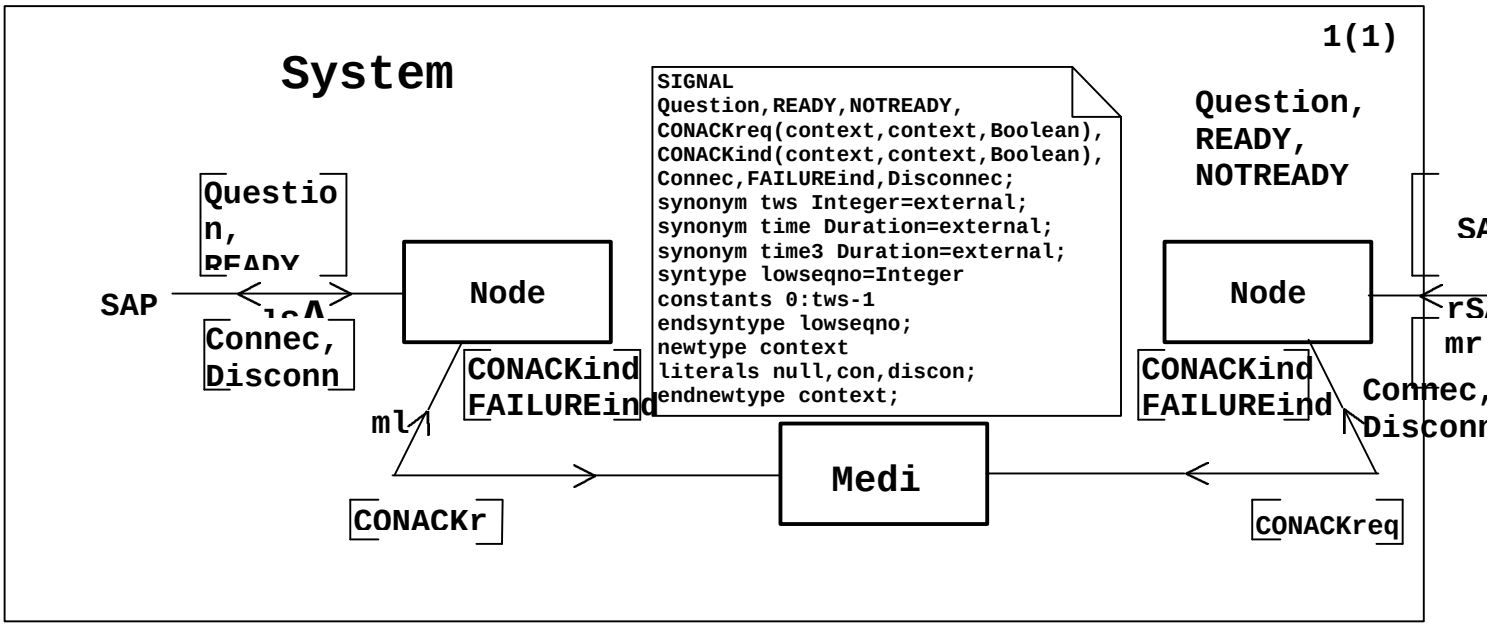

Figure 3 – Two nodes A and B communicating through a channel

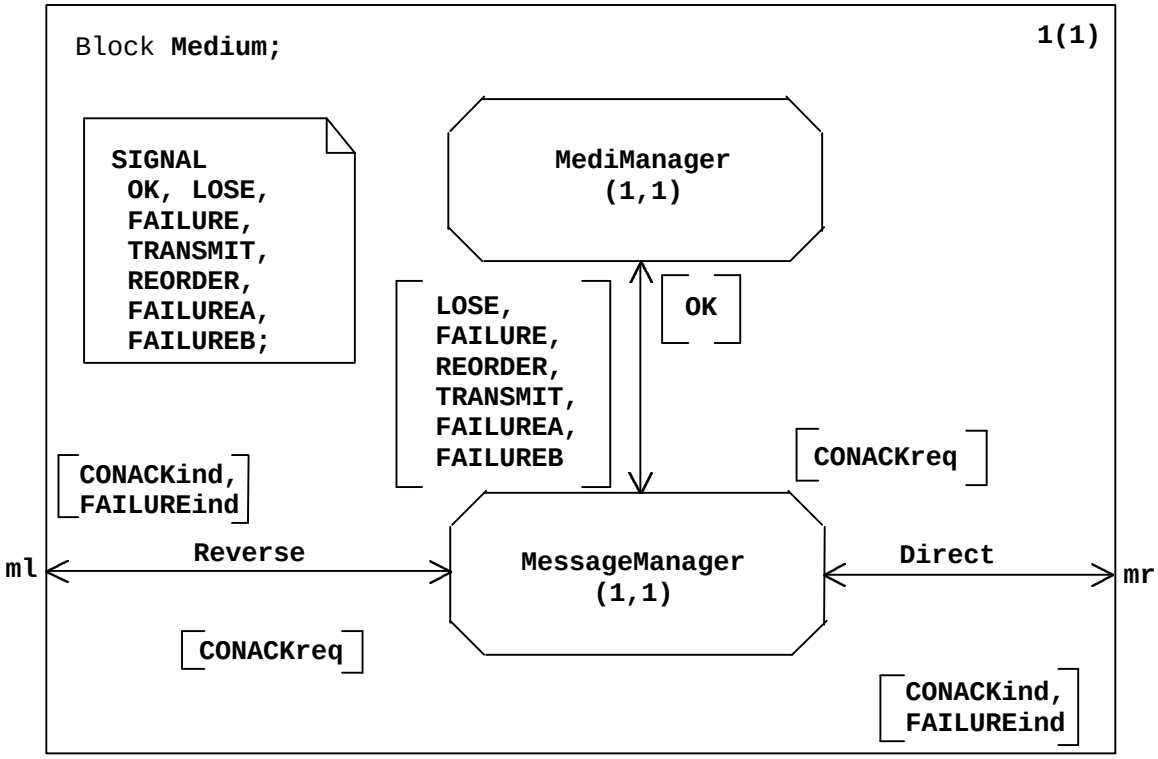

Figure 4 – Block “Medium” describes the channel

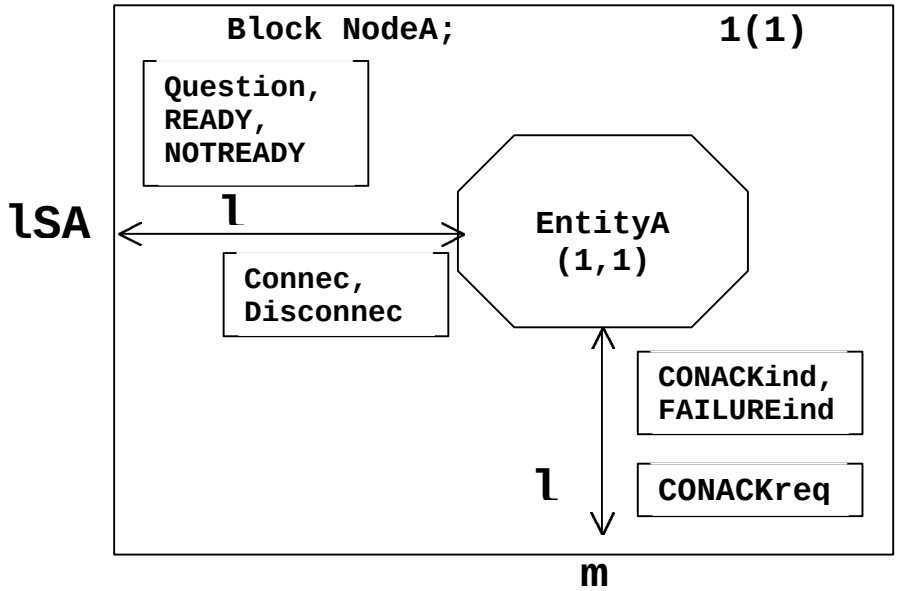

Figure 5 – Node A in detail