12

An Algebraic-Temporal Specification of a CSMA/CD-Protocol

Mohamed Jmaiel
Technische Universität Berlin
Sekr. FR 5-13, Franklinstr. 28/29, D-10587 Berlin
Tel. +49 30 314 21763, Fax. +49 30 314 73623.
email: mojm@cs.tu--berlin.de

Abstract

This paper presents a formal development of a CSMA/CD (Carrier Sense, Multiple Access with Collision Detection) protocol. Using a combination of temporal logic and algebraic specifications we describe the message layout and the behavioral aspects of the protocol in a unified framework. We benefit from the deduction system of temporal logic to establish safety and liveness properties of the protocol.

Keywords

Communication protocols, formal methods, stepwise development, refinements, algebraic specifications, and temporal logic.

1 INTRODUCTION

A distributed system consists of a set of agents that execute independently and interact with each other in order to exchange data. The programs that are responsible for the realization of a reliable data exchange within a distributed system are called *communication protocols*. Usually, communication protocols exhibit extremely intricate behavior, since they must cope with the possibility of failures in the physical components. Due to their complexity, the development of protocols is considered as a hard task and should follow rigorous formal techniques, in order to ensure correct implementations. In this context, the formal specification of protocols is of particular importance, since it is the basis of a correct implementation. In this paper, we deal with the specification of a CSMA/CD protocol at a very high level of abstraction. For this purpose, we make use of a recently proposed specification and verification technique based on a unification of algebraic and temporal specifications (Jmaiel & Pepper 1994).

Many different formal languages have been developed and applied to the description of protocols. The most important approaches are finite state machines, CSP (Hoare 1985), CCS (Milner 1980), Petri nets (Petri 1962), and temporal logic. However, the majority of the developed languages consider only the description of the behavioral aspect of a protocol, whereas the data aspect of protocols is treated

P. Dembiński et al. (eds.), *Protocol Specification, Testing and Verification XV*

separately using other specification methods such as Z (Abrial 1985), VDM (Bjørner & Jones 1987), or algebraic specifications (Ehrig & Mahr 1985). From a methodological point of view, this separation may lead to some difficulties during the development process, since a suitable notion of composition of the specification languages is needed in order to be able to derive the correctness of the protocol implementation from the correctness of the implementations of the data part and the behavioral part. On the other hand, the user has to follow two development processes based on different kinds of transformation rules, rather than one process based on one sort of rules. In our opinion, a formal specification language for protocols should be complete in the sense that it covers all relevant aspects of a protocol.

The integration of temporal logic with algebraic specifications provides a unified framework in which data aspects as well as behavioral aspects can be handled at a high level of abstraction. Moreover, this language offers two advantages. First, a formal temporal proof system can be applied to verify safety and liveness properties. Second, the application of a modular temporal logic allows the stepwise development of design specifications from requirement specifications using compositions and refinements.

This paper is organized as follows. The next section describes the aspects considered in the specification of protocols. In Section 4 the temporal logic used here is defined. Section 3 presents the algebraic framework we use for the specification of the CSMA/CD protocol. In Section 5 we give a requirement specification describing the service provided by the CSMA/CD protocol. Based on a specification of the transmission medium, in Section 6 we develop step by step an implementation of the protocol.

2 HOW TO SPECIFY

In our approach a distributed system is modeled by a family of interacting *agents* that represent the logical units of the system, e.g. sender, receiver, or transmission medium. One of the main characteristics of an agent is that it does not run in isolation, but that it exchanges data with its environment via unidirectional *channels*. A channel is considered as an abstract interface of an agent with its environment. So an agent may be viewed as an entity that receives and sends data on its input and output channels, respectively. Basically, this representation corresponds to the CSP model proposed by Hoare (Hoare 1985).

A network of agents is formed by linking some input channels of some agents to some output channels of other agents in one-to-one manner. A network may be graphically represented as in Figure 1.

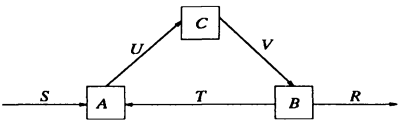


Figure 1 Graphical Representation of a Network.

However, it is important not to confuse channels and physical transmission lines; channels are conceptual representations of the communications between the agents. Semantically, channels are considered as streams of data elements. Thus, they are absolutely reliable, in contrast to physical transmission media which, in general, may lose, duplicate, or permute messages. Hence, physical transmission media need to be modeled as active agents.

According to a given network, we specify the behavior of the agents by a set of temporal formulas expressing *safety* and *liveness* properties of the system. However, we do not consider any details relating

to the internal structure of the agents. That is, an agent is viewed as a black box that passes messages from its input to its output channels. From this point of view, the specification of a system is restricted to the description of its observable input-output behavior, characterized by a relation between the communication actions occurring during a system run; this relation is often called *causality* relation.

In a network of agents, the basic actions that may occur are transmissions of messages; their occurrences are expressed by predicates such as

$[A \textbf{ rcv } m \textbf{ on } S]$	The agent A receives the message m on the channel S.
$[U \textbf{ xmt } m]$	The channel U transmits the message m.
$[B \textbf{ snd } m \textbf{ on } T]$	The agent B sends the messages m on the channel T.

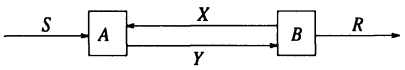
These predicates constitute the atomic formulas of the specification language. We should mention that here we are considering networks with *strongly coupled communication*: such networks are characterized by the equalities (referring to the network in Figure 1)

$$[A \textbf{ snd } m \textbf{ on } U] \equiv [U \textbf{ xmt } m] \equiv [C \textbf{ rcv } m \textbf{ on } U]$$

3 ALGEBRAIC-TEMPORAL SPECIFICATIONS

Based on the idea that agents may be represented by stream-processing functions and channels represented by streams (see e.g. (Broy 1988)), one can embed temporal formulas in the framework of algebraic specifications. We follow the schema proposed in (Jmaiel & Pepper 1994) for presenting protocol specifications.

SPECIFICATION ≪ *name* ≫
  IMPORT ≪ *other specifications* ≫
  SIGNATURE ≪ *collection of agents and their functionalities, such as* ≫
    FUN A : *stream* × *stream* → *stream*
    FUN B : *stream* → *stream* × *stream*
  NETWORK ≪ *a communication network for the given agents, such as* ≫



  PROPERTIES ≪ *temporal formulas describing the behavior of the agents* ≫

The import list makes some sorts, operations, and agents defined in other specifications available in the current specification. This is very useful for describing protocols, since repetitions are avoided and specifications become more structured.

Actually, the graphical representation of the network is merely a syntactical sugaring for the temporal formulas. However, such a graphical representation can be characterized algebraically by a specific system of equations. The above network is therefore an abbreviation of the following extension of the property part:

PROPERTIES $\forall\, S, X, Y, R$: *stream*

$$A(S,X) = Y \wedge B(Y) = (X,R) \Rightarrow \ll \textit{temporal properties} \gg$$

This entails that these specifications usually introduce and characterize a certain set of agents, whereas the names of the streams are bound and thus exchangeable.

In the above framework one can also describe the layout of messages traveling on the channels as well as the operations available to the agents for treating them. In the context of protocols, the signature may comprise the sorts of messages to be transmitted within the system as well as operations such as splitting messages into packets, selecting fragments from packets (e.g. destination address, sequence number, or error detection code), and building complete messages from individual packets. Generally, the classical equational approach (e.g. (Ehrig & Mahr 1985)) is sufficiently powerful to formulate the properties of such operations.

In the following we will treat the data type aspect of protocols only marginally. Instead, we concentrate on issues conserning the specification of temporal properties of protocols.

4 A MODULAR TEMPORAL LOGIC

Temporal logic is a simple and elegant extension of propositional logic (predicate logic in the case of first-order temporal logic), yet it is powerful enough to express interesting properties of distributed systems such as *safety* and *liveness* properties. Temporal formulas are constructed from atomic formulas by applying temporal and boolean operators. Temporal operators refer to the past and future. The future operators include the unary operators $\bigcirc$ (*nexttime*), $\Box$ (*always*), and $\Diamond$ (*eventually*), and the binary operator **before**. Usually, the past operators include a symmetric counterpart to each of the future operators (see, e.g. (Lichtenstein, Pnueli & Zuck 1985)). Here we make use only of the unary operators $\bullet$ (*previous*) and $\blacklozenge$ (*eventually in the past*).

Indexing Temporal Operators: Temporal logic in its elementary form does not support modular specification. The reason is that temporal formulas are global, i.e. they refer to all components of the system specified (see e.g. (Barringer, Kuiper & Pnueli 1984)). Therefore, one cannot specify a component independently of its environment. In order to overcome this difficulty, we index every temporal operator with those channels to which it should apply. Doing so, a temporal formula refers to a specific component (module) rather than to all components.

Let M be a set of channels and P be a temporal formula. We say M is at a time-point active if a transmission action is taking place on one of the elements of M. Informally, our temporal operators have the following meaning*. A formal definition of these operators is given in the Appendix.

$\triangle_M P$ P holds "now" on M, i.e. M is active and P holds.
$\bigcirc_M P$ P holds at the next time-point at which M will be active.
$\Box_M P$ P holds at every future time-point at which M is active (including the present).
$\Diamond_M P$ P holds at some future time-point at which M is active (including the present).
$\bullet_M P$ P held at the last time-point at which M was active.
$\blacklozenge_M P$ P held at some past time-point at which M was active (including the present).

*To aid understanding, our definition of the operators is based on a concept of "time", but actually the underlying notion is that of "causal dependency".

$P\ {}_M\mathbf{before}_N\ Q$ there is a point in time on M (possibly "now") where P holds that strictly precedes the first point in time where Q holds on N.

To these operators we add a unary operator $\Diamond\!\!\!\!+$ (*later* or *strict eventually*) that will often be used in our specifications; it is defined as

$$\Diamond\!\!\!\!+_M\ P \stackrel{\text{def}}{\Longleftrightarrow} \bigcirc_M \Diamond_M P$$

We omit the index if it includes all channels of the system. In this case temporal operators refer implicitly to the whole system behavior and thus equivalent to the classical ones. Moreover, if R is a channel name, we write $\Diamond_R P$ instead of $\Diamond_{\{R\}} P$.

If we combine temporal operators with predicates that refer to the activity of transmitting messages we obtain formulas like:

$\Diamond_S [S\ \mathbf{xmt}\ m]$	Eventually the channel S transmits the message m.
$\Box \Diamond_S [S\ \mathbf{xmt}\ m]$	The message m will be infinitely often transmitted by S.
$[T\ \mathbf{xmt}\ m] \Rightarrow \bigcirc_S [S\ \mathbf{xmt}\ m]$	If the channel T transmits a message m, then the next message transmitted on the channel S will be m.
$\triangle_R [A\ \mathbf{snd}\ m]$	The agent A transmits the message m on the channel R. This formula is equivalent to $[A\ \mathbf{snd}\ m\ \mathbf{on}\ R]$.

It is important not to confuse the formula $\Box [S\ \mathbf{xmt}\ m]$ with $\Box_S [S\ \mathbf{xmt}\ m]$. The former means that the activity "S transmits m" takes place at every future time-point, while the formula $\Box_S [S\ \mathbf{xmt}\ m]$ says that whenever S is transmitting, it transmits the message m.

5 THE CSMA/CD PROTOCOL

The CSMA/CD (Carrier Sense, Multiple Access with Collision Detection) protocol provides a medium access method intended for use in local area networks. Generally, local area networks may present an access conflict when transmitting frames on the physical medium because all connected stations share a common transmission line. Before initiating a transmission, a station asks whether the medium is free or not; if the medium is free a frame transmission is initiated, otherwise the transmission is delayed. However, two stations may test the medium at the same time, and consequently they may transmit frames simultaneously. If two stations attempt to transmit frames at the same time, then these frames collide on the transmission line, and consequently, they get (partially) lost. The main task of the CSMA/CD protocol is to provide a reliable transmission of frames between the stations of a local area network.

The specification presented here is based on the standard definition (Ame 1985). An overview of the protocol structure and its relationship to the OSI reference model is shown in Figure 2. Following the standard definition, the function intended for the data link layer is accomplished by two sublayers: the Logical Link Control (LLC) and the Media Access Control (MAC). In considering the CSMA/CD protocol, the MAC sublayer plays the most important role in providing the facilities of sending and receiving frames. Nevertheless, the services provided by the physical layer are the basis for the MAC sublayer to accomplish its function. Hence, in our development we concentrate on the behavior of these two layers: the MAC and the physical layers. First, we give a separate specification to each of them. Then

we develop a specification which describes how the MAC sublayer realizes its services with regard to the services provided by the physical layer.

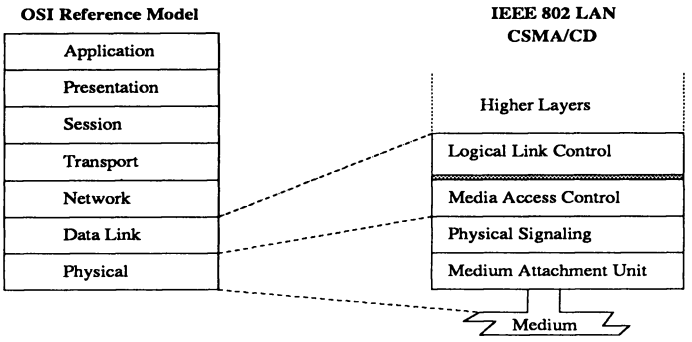


Figure 2 LAN Standard Relationship to the OSI Model

5.1 Specification of the CSMA/CD Protocol

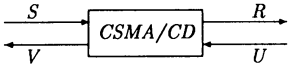
In this section we describe the services provided by the MAC sublayer, that is, the services provided at the interfaces between the LLC and MAC sublayers. For simplicity, we consider a local area network with two stations. Following the standard definition, the MAC layers together with the lower layers and the physical medium provide a bidirectional transmission of messages† to the LLC entities. Actually, the transmission of frames is not always successful; after a transmission request the LLC sublayer is informed as to whether its data have been successfully transmitted or not. Moreover, the MAC sublayer may also pass illegal data to the LLC, which may consist of corrupted or incomplete messages. However, we do not consider the case of failures in our specifications, rather investigate an idealized case in which the transmission is error free. Hence, we view the system as an agent (called CSMA/CD) that provides a reliable bidirectional transmission of messages between the LLC entities. A reliable transmission is characterized by the following three properties:

- messages are not created, i.e. every sent message must have been previously received,
- messages are not lost, i.e. every message received will eventually be sent,
- messages are sent in the same order as they have been received.

These properties are formulated in the following specification by the theorems $Prop_1$, $Prop_2$, and $Prop_3$, respectively. The specification mainly describes the behavioral aspects of the system, whereas the concepts concerning the message layout are treated in another specification, `Message`. Such an algebraic specification describes the type of messages exchanged at the interfaces between the LLC and the MAC layers. From this specification we need only import the sort of legal messages, *msg*.

†Messages correspond to the data passed between the LLC sublayer and the MAC sublayer.

SPECIFICATION **CSMA/CD**
IMPORT **Message** ONLY *msg*
SIGNATURE
FUN $CSMA/CD : stream[msg] \times stream[msg] \rightarrow stream[msg] \times stream[msg]$
NETWORK



PROPERTIES $\forall m, n : msg$
AXM $(Prop_1)$: $[R\ \mathbf{xmt}\ m] \Rightarrow \blacklozenge [S\ \mathbf{xmt}\ m]$
AXM $(Prop_2)$: $[S\ \mathbf{xmt}\ m] \Rightarrow \Diamond [R\ \mathbf{xmt}\ m]$
AXM $(Prop_3)$: $[R\ \mathbf{xmt}\ m] \wedge \blacklozenge [R\ \mathbf{xmt}\ n] \Rightarrow \blacklozenge ([S\ \mathbf{xmt}\ m] \wedge \blacklozenge [S\ \mathbf{xmt}\ n])$

analogously for U and V

We should stress here that this specification corresponds to the properties listed above only under the assumption that the messages on the streams are distinct. Without this assumption one cannot guarantee the desired behavior of the protocol; for example, a computation that sends only one message for two identical messages received would satisfy the formula $Prop_2$, if one allows that identical messages may be transmitted on S.

This problem is not specific for the specification presented here. Sistla et al. (Sistla, Clarke, Francez & Meyer 1984) have proved that it is impossible to give a temporal specification of an asynchronous buffer without the assumption of distinct messages. Similar results have been achieved by Koymans (Koymans 1992). However, this assumption is not restrictive, since one can impose the distinction of messages by annotating data elements with abstract time-stamps. Doing so, one ensures that messages are distinct even when they have the same data content.

In our approach, we introduce time-stamps only at the conceptual level, as an issue for the technical difficulty mentioned above. This entails that they are hidden for agents, i.e. the behavior of an agent does not depend on time-stamps. Indeed, their introduction at the syntactical level depends on wether the protocol modules (in the implementation) need to test time-stamps or not. Accordingly, we assume that messages are annotated with time-stamps. We denote by $m \not\sim n$ the fact that the messages m and n have distinct time-stamps. Further, we assume that messages on the streams S, R, U, and V are respectively annotated with distinct time-stamps. This can be formulated by temporal formulas: for example for S:

(Unq) : $[S\ \mathbf{xmt}\ m] \wedge \blacklozenge [S\ \mathbf{xmt}\ n] \Rightarrow m \not\sim n$

This formula says that if S transmits a message m and then later another message n, then m and n must be distinctly annotated.

5.2 Transmission Medium

The transmission medium consists of the physical layers together with the actual physical medium. It is considered as an agent M that provides an unreliable bidirectional transmission of frames between the MAC entities. The unreliability is caused by the possibility of destroying messages whenever they collide.

The messages passed at the interface between the physical layer and the MAC sublayer are called frames. Apart from a data field, a frame includes fields concerning, e.g. destination and source addresses,

frame length, and frame check sequence, needed to verify the validity of a frame. All these topics can, of course, be specified algebraically. We assume the existence of a specification, **Frame**, describing the layout of frames. For the description of the behavioral aspects of the medium, we import the sort of legal frames (*frame*), the constant symbol *coll*, which is used to indicate the occurrence of a collision, and the symbol *ackn*, needed to confirm a successful transmission of a frame. In practice, a transmission of a frame is automatically acknowledged if the collision-detect signal remains inactivate during the transmission of the frame.

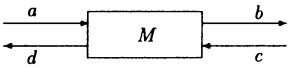
The following specification describes the service provided by the transmission medium to the MAC entities. Based on the standard definition, the medium should fulfill the following properties:

- A characteristic property of the underlying network (Bus-System) is that if a frame is put into the medium, then either it comes through undamaged or it collides with another frame. This property is formulated by the temporal formula M_1 below.
- One of the main tasks of the medium is to detect a collision and to signal it immediately by sending the *coll* signal to the participating users. Hence, the property M_2 states that if a collision has occurred, then the constant *coll* is sent on the channels b and c, respectively.
- Legal frames may be lost but they should not be created. Hence, the property M_3 states that the medium can only send what it has received.
- The property M_4 ensures that the medium cannot confirm a frame before it has been successfully delivered, whereas property M_5 states that a successful transmission is acknowledged immediately after delivering a frame.

In the following specification, the predicate *Collision* is an abbreviation for a temporal formula expressing that a collision has arisen. A collision arises if two frames are put into the medium simultaneously. In our linear-time logic, simultaneous occurrence of two actions is modeled by a non-deterministic interleaving. Thus, a collision arises if a frame is sent to the medium before an already received one has been sent. Formally:

$$\mathit{Collision} \equiv [M\ \mathbf{rcv}\,] \wedge \bullet_M [M\ \mathbf{rcv}\,]$$

SPECIFICATION **Medium**
IMPORT **Frame** ONLY *frame coll ackn*
SIGNATURE
FUN M : $\mathit{stream}[\mathit{frame}] \times \mathit{stream}[\mathit{frame}] \rightarrow \mathit{stream}[\mathit{frame}] \times \mathit{stream}[\mathit{frame}]$
NETWORK



PROPERTIES $\forall p, q : \mathit{frame} \backslash \{\mathit{coll}, \mathit{ackn}\}$
AXM (M_1) : $\triangle_a [M\ \mathbf{rcv}\ p] \Rightarrow \mathit{Collision} \vee \bigcirc_b [M\ \mathbf{snd}\ p]$
AXM (M_2) : $\mathit{Collision} \Rightarrow \bigcirc_b [M\ \mathbf{snd}\ \mathit{coll}] \wedge \bigcirc_d [M\ \mathbf{snd}\ \mathit{coll}]$
AXM (M_3) : $\triangle_b [M\ \mathbf{snd}\ p] \Rightarrow \bullet_a [M\ \mathbf{rcv}\ p]$
AXM (M_4) : $\triangle_d [M\ \mathbf{snd}\ \mathit{ackn}] \Rightarrow [M\ \mathbf{rcv}\ p]\ {}_a\mathbf{before}_b\ [M\ \mathbf{snd}\ p]$
AXM (M_5) : $\triangle_b [M\ \mathbf{snd}\ p] \Rightarrow [M\ \mathbf{snd}\ \mathit{ackn}]\ {}_d\mathbf{before}_b\ [M\ \mathbf{snd}\ q]$

analogously for direction $c \rightarrow d$

This specification describes the transmission of frames from the channel a to the channel b. The transmission from c to d works in the same way. Therefore, we should add to the above specification similar axioms to M_1, M_3, M_4, and M_5 referring to the channels c and d. All we need to do is replace in these axioms the occurrences of a, b and d by c, d and b, respectively.

As we will see in the next section, these conditions guarantee the proper functioning of the CSMA/CD protocol.

6 FORMAL DERIVATION OF THE PROTOCOL SPECIFICATION

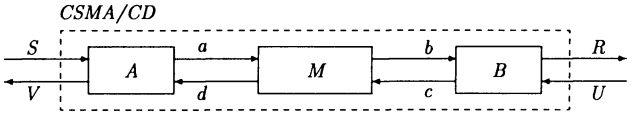
In Section 5.1 we presented a specification that describes the services provided by the CSMA/CD protocol, without giving any internal details of the system. In this section we take some design decisions and show how these services are provided, based on the services of the transmission medium.

In our development process we follow principles applied in the KorSo-Project[‡] for the realization of algebraic specifications (see e.g. (Pepper & Wirsing 1994)). For a given protocol specification SP_1 we develop step by step a specification SP_2 that implements it, i.e. its external behavior is the same, but it has a more detailed internal structure.

As soon as we decide to implement the CSMA/CD protocol we have to take into consideration the unreliability of the transmission medium. Thus, the main task we have to accomplish is to augment the system originally consisting of the unreliable medium such that externally we get a reliable transmission of messages. Therefore, we add two agents A and B which correspond to the protocol entities of the MAC sublayers. The next, even more important, step is to design the behavior of the newly introduced agents. We proceed as follows: we try to perform the correctness proof; in doing so we determine which properties (about the behavior of the new agents) are needed in order to perform the proof.

The proof obligations are $(Prop_1)$, $(Prop_2)$ and $(Prop_3)$ in the specification presented in Section 5.1. We start with the following specification, which does not include any requirements on the behavior of the new agents A and B, but includes all requirements made on the behavior of the medium:

SPECIFICATION **CSMA/CD-Implementation**
IMPORT **Medium**
IMPORT **Message** ONLY *msg*
SIGNATURE
FUN A : $stream[msg] \times stream[frame] \rightarrow stream[msg] \times stream[frame]$
FUN B : $stream[msg] \times stream[frame] \rightarrow stream[msg] \times stream[frame]$
NETWORK



PROPERTIES ≪ *will be developed later* ≫

[‡]KorSo "Korrekte Software" is sponsored by the German Ministry of Research and Technology.

This specification will be enriched by properties about the behavior of the agents A and B until the correctness proof is performed. The resulting final specification is then an implementation of the specification presented in Section 5.1.

In communicating with the transmission medium, the agents A and B have to deal with frames, while in communicating with the LLC entities they exchange messages. Thus, messages must be transformed to frames, and vice versa. The transformations are internal routines that should be performed by both agents A and B. For simplicity, we assume that a message is encapsulated within one frame, and that a frame is decapsulated to one message. So, we enrich to the system specification by two operations f, for transforming messages into frames, and g, for transforming frames into messages. The main property that we are interested in is that encapsulating a message and then decapsulating it yields the original message. Hence, we extend the system specification in the following way:

ENRICH **CSMA/CD-Implementation** BY
  SIGNATURE
    FUN $f : msg \rightarrow frame$
    FUN $g : frame \rightarrow msg$
  PROPERTIES $\forall m : msg$
    AXM $: g(f(m)) = m$

In order to ensure the consistency of our specifications we have to assume that the functions f and g preserve the time-stamps of their arguments.

6.1 Step 1: Proof of Safety Property: Prop_1

We try to establish the safety property saying that frames cannot be created. It turns out that we need to add requirements to the agents A and B, which guarantee that they only send what they have received.

Proof of : $[R\,\mathbf{xmt}\,m] \Rightarrow \blacklozenge\,[S\,\mathbf{xmt}\,m]$

	$[R\,\mathbf{xmt}\,m]$	
$\Rightarrow$	$\bullet_b\,[B\,\mathbf{rcv}\,f(m)]$	[by axiom B_1 below]
$\Rightarrow$	$\blacklozenge_b\,[M\,\mathbf{snd}\,f(m)]$	[by temporal logic]
$\Rightarrow$	$\blacklozenge_b \bullet_a\,[M\,\mathbf{rcv}\,f(m)]$	[by axiom M_3]
$\Rightarrow$	$\blacklozenge_b \bullet_a\,[A\,\mathbf{snd}\,f(m)]$	[by temporal logic]
$\Rightarrow$	$\blacklozenge_b \bullet_a \bullet_S\,[A\,\mathbf{rcv}\,g(f(m))]$	[by axiom A_1 below]
$\Rightarrow$	$\blacklozenge\,[S\,\mathbf{xmt}\,m]$	[by temporal logic]

q.e.d

We have to enrich the properties of our system by the following two axioms:

ENRICH **CSMA/CD-Implementation** BY
  PROPERTIES $\forall p : frame, m : msg$
    AXM (A_1) : $\triangle_a\,[A\,\mathbf{snd}\,p] \Rightarrow \bullet_S\,[A\,\mathbf{rcv}\,g(p)]$
    AXM (B_1) : $\triangle_R\,[B\,\mathbf{snd}\,m] \Rightarrow \bullet_b\,[B\,\mathbf{rcv}\,f(m)]$

Note that the axiom A_1 requires that the agent A can send only the last received frame on the channel S. However, a weak version of this axiom, such as $\triangle_a\,[A\ \mathbf{snd}\ p] \Rightarrow \blacklozenge_S\,[A\ \mathbf{rcv}\ g(p)]$, would be sufficient to perform the proof of the safety property. We have chosen the strong version for two reasons. On the one hand, the property A_1 guarantees that the order of messages is preserved by A. On the other hand, in connection with axiom A_3 (see Section 6.2), we ensure that the agent A can only start the transmission of a frame if the previous one has been successfully delivered.

6.2 Step 2: Proof of Liveness Property: $\mathtt{Prop}_2$

In this section we deal with the proof of the property $\mathtt{Prop}_2$, which is a liveness property stating that every frame on S will be eventually delivered on the stream R. In considering the transmission from S to R, the realization of this behavior is completely the responsibility of the agent A, which should repeat the transmission of a frame untill it has been delivered to B.

Proof of : $[S\ \mathbf{xmt}\ m] \Rightarrow \Diamond\,[R\ \mathbf{xmt}\ m]$

	$[S\ \mathbf{xmt}\ m]$	
$\Rightarrow$	$\Diamond_a\,[A\ \mathbf{snd}\ f(m)]$	[by axiom A_2 below]
$\Rightarrow$	$\Diamond_a\,([A\ \mathbf{snd}\ (f(m)] \wedge \neg \mathit{Collision})$	[by theorem *nocoll* below]
$\Rightarrow$	$\Diamond_a\,\bigcirc_b\,[M\ \mathbf{snd}\ f(m)]$	[by axiom M_1]
$\Rightarrow$	$\Diamond_b\,[M\ \mathbf{snd}\ f(m)]$	[by temporal logic]
$\Rightarrow$	$\Diamond_b \Diamond\,[R\ \mathbf{xmt}\ g(f(m)]$	[by axiom B_2 below]
$\Rightarrow$	$\Diamond\,[R\ \mathbf{xmt}\ m]$	[by temporal logic]

q.e.d.

This proof requires the extension of the specification `CSMA/CD-Implementation` by the following liveness properties A_2 and B_2; these state that the agents A and B should eventually send what they have received.

ENRICH `CSMA/CD-Implementation` BY
  PROPERTIES $\forall p : \mathit{frame},\ m : \mathit{msg}$
    AXM (A_2) : $[S\ \mathbf{xmt}\ m] \Rightarrow \Diamond_a\,[A\ \mathbf{snd}\ f(m)]$
    AXM (B_2) : $\triangle_b\,[B\ \mathbf{rcv}\ p] \Rightarrow \Diamond\,[R\ \mathbf{xmt}\ g(p)]$

Further, we have to prove the property *nocoll*, stating that if the agent A starts the transmission of a frame, then eventually it reaches a state, where no collision occurs. This entails that the frame will be delivered intact.

THM (nocoll) : $\triangle_a\,[A\ \mathbf{snd}\ p] \Rightarrow \Diamond_a\,([A\ \mathbf{snd}\ p] \wedge \neg \mathit{Collision})$

Proof of: *nocoll*
We perform the proof this proof by contradiction, beginning with the following assumption:

$(*)$: $[A\ \mathbf{snd}\ m] \wedge \Box_a\,([A\ \mathbf{snd}\ m] \Rightarrow \mathit{Collision})$

Then we can perform the following deduction:

	$[A\ \mathbf{snd}\ p] \wedge \Box_a([A\ \mathbf{snd}\ p] \Rightarrow \mathit{Collision})$	[by (*)]
$\Rightarrow$	$[A\ \mathbf{snd}\ p] \wedge \mathit{Collision} \wedge \Box_a([A\ \mathbf{snd}\ p] \Rightarrow \mathit{Collision})$	[by temporal logic]
$\Rightarrow$	$[A\ \mathbf{snd}\ p] \wedge \bigcirc_d[M\ \mathbf{snd}\ \mathit{coll}] \wedge \Box_a([A\ \mathbf{snd}\ p] \Rightarrow \mathit{Collision})$	[by axiom M_2]
$\Rightarrow$	$[A\ \mathbf{snd}\ p] \wedge \bigcirc_a[A\ \mathbf{snd}\ p] \wedge \Box_a([A\ \mathbf{snd}\ p] \Rightarrow \mathit{Collision})$	[by axiom A_3 below]
$\Rightarrow$	$\Box_a[A\ \mathbf{snd}\ p] \wedge \Box_a([A\ \mathbf{snd}\ p] \Rightarrow \mathit{Collision})$	[by temporal logic]
$\Rightarrow$	$\Box_a\ \mathit{Collision}$	[by temporal logic]

This contradicts axiom A_4 below!
q.e.d.

The proof of this property requires that the agent A retransmits a frame whenever it receives the constant *coll* indicating the occurrence of a collision. Further, it is required that a collision cannot occur continuously. So it is necessary to augment the specification `CSMA/CD-Implementation` with the following axioms:

ENRICH `CSMA/CD-Implementation` BY
  PROPERTIES $\forall p$: *frame*
    AXM (A_3) : $\triangle_a[A\ \mathbf{snd}\ p] \wedge \bigcirc_d[A\ \mathbf{rcv}\ \mathit{coll}] \Rightarrow \bigcirc_a[A\ \mathbf{snd}\ p]$
    AXM (A_4) : $\neg\Box_a\ \mathit{Collision}$

Theoretically, it is possible that a collision occurs infinitely such that one cannot get frames through the medium. In order to establish the liveness property, it is therefore necessary to first assume that a collision cannot occur continuously. In practice, after a maximal number of attempts the transmission is abandoned and the LLC layer is informed that the transmission of its data has failed.

6.3 Step 3: Proof of Safety Property: $\mathtt{Prop}_3$

The property M_2 ensures that the transmission medium delivers frames in the same order as they have been received. On the other hand, the property of preserving the order of frames is also satisfied by both agents A and B. Therefore, we deduce immediately that the whole system satisfies this property.

Proof of: $[R\ \mathbf{xmt}\ m] \wedge \blacklozenge[R\ \mathbf{xmt}\ n] \Rightarrow \blacklozenge([S\ \mathbf{xmt}\ m] \wedge \blacklozenge[S\ \mathbf{xmt}\ n])$

	$[R\ \mathbf{xmt}\ m] \wedge \blacklozenge[R\ \mathbf{xmt}\ n]$	
$\Rightarrow$	$\triangle_R[B\ \mathbf{snd}\ m] \wedge \blacklozenge_R[B\ \mathbf{snd}\ n]$	[by temporal logic]
$\Rightarrow$	$\blacklozenge(\triangle_b[M\ \mathbf{snd}\ g(m)] \wedge \blacklozenge_b[M\ \mathbf{snd}\ g(n)])$	[by Order_B below]
$\Rightarrow$	$\blacklozenge\blacklozenge(\triangle_a[A\ \mathbf{snd}\ g(m)] \wedge \blacklozenge_a[M\ \mathbf{snd}\ g(n)])$	[by Order_M below]
$\Rightarrow$	$\blacklozenge\blacklozenge\blacklozenge([S\ \mathbf{xmt}\ f(g(m))] \wedge \blacklozenge[S\ \mathbf{xmt}\ f(g(n))])$	[by Order_A below]
$\Rightarrow$	$\blacklozenge([S\ \mathbf{xmt}\ m] \wedge \blacklozenge[S\ \mathbf{xmt}\ n])$	[by temporal logic]

q.e.d.

In the folowing we prove the property stating that the agent B preserves the order of the received messages:

THM (Order_B) : $(\triangle_R[B\ \mathbf{snd}\ m] \wedge \blacklozenge_R[B\ \mathbf{snd}\ n]) \Rightarrow \blacklozenge(\triangle_b[B\ \mathbf{rcv}\ g(m)] \wedge \blacklozenge_b[B\ \mathbf{rcv}\ g(n)])$

$\quad\quad \triangle_R\,[B\ \mathbf{snd}\ m] \wedge \blacklozenge_R\,[B\ \mathbf{snd}\ n]$
$\Rightarrow \quad \bullet_b\,[B\ \mathbf{rcv}\ g(n)] \wedge \blacklozenge \bullet_b\,[B\ \mathbf{rcv}\ g(n)]$ [by axiom B_1]
$\Rightarrow \quad \bullet_b\,([B\ \mathbf{rcv}\ g(m)] \wedge \blacklozenge_b\,[B\ \mathbf{rcv}\ g(n)])$ [by temporal logic]
$\Rightarrow \quad \blacklozenge\,(\triangle_b\,[B\ \mathbf{rcv}\ g(m)] \wedge \blacklozenge_b\,[B\ \mathbf{rcv}\ g(n)])$ [by temporal logic]
q.e.d.

The properties $Order_A$ and $Order_M$ can be formulated and proved in a similar way.

6.4 Step 4: Ensuring Consistency

A necessary condition for the implementability of the agents A and B is the *consistency* of their specifications. This is no problem for $A_1 - A_4$; we can easily develop an implementation that meets all these requirements. The case is slightly different for the specification of B. According to the assumption claiming that messages on the stream R are all distinct, the proper functioning of the agent B can only be guaranteed if the medium M does not duplicate frames on the stream b. Otherwise, we may obtain the following situation, where M sends a duplicate of a frame which has already been transmitted on R. Then we deduce:

$\quad\quad \exists p : \triangle_R\,[B\ \mathbf{snd}\ g(p)] \wedge \Diamond_b\,[M\ \mathbf{snd}\ p]$
$\Rightarrow \quad \exists p : \triangle_R\,[B\ \mathbf{snd}\ g(p)] \wedge \Diamond_R\,[B\ \mathbf{snd}\ g(p)]$ [by axiom B_2]
$\Rightarrow \quad \exists p : g(p) \not\sim g(p)$ [by *Unq*]
$\Rightarrow \quad \exists p : p \not\sim p$ [by property of g]
Contradiction!
q.e.d.

For this reason we should ensure that the Medium does not duplicate frames. If we consider the properties M_2, M_4, and M_5, then we can deduce that the medium could not duplicate frames, provided that the agent A would stop the retransmission of a frame as soon as the transmission of the frame is acknowledged. However, the properties $A_1 - A_4$ allow that the agent A could continue to retransmit a frame, even if it has been already delivered to B. This may lead to delivering a frame several times. Hence, we have to establish the following property stating that the frames on the stream b are distinctly annotated.

THM(M_6) : $\quad \triangle_b\,[M\ \mathbf{snd}\ p] \wedge \Diamond_b\,[M\ \mathbf{snd}\ q] \Rightarrow p \not\sim q$

Proof of M_6:

$\quad\quad \triangle_b\,[M\ \mathbf{snd}\ p] \wedge \Diamond_b\,[M\ \mathbf{snd}\ q]$
$\Rightarrow \quad \triangle_b\,[M\ \mathbf{snd}\ p] \wedge \Diamond_d\,([M\ \mathbf{snd}\ ackn] \wedge \Diamond_b\,[M\ \mathbf{snd}\ q])$ [by axiom M_5]
$\Rightarrow \quad \triangle_b\,[M\ \mathbf{snd}\ p] \wedge \Diamond_d\,([M\ \mathbf{snd}\ ackn] \wedge \Diamond_a\,[M\ \mathbf{rcv}\ q])$ [by axiom M_4]
$\Rightarrow \quad \bullet_a\,[M\ \mathbf{rcv}\ p] \wedge \Diamond_d\,([M\ \mathbf{snd}\ ackn] \wedge \Diamond_a\,[M\ \mathbf{rcv}\ q]$ [by axiom M_3]
$\Rightarrow \quad \Diamond\,(\blacklozenge_a\,[A\ \mathbf{snd}\ p] \wedge [A\ \mathbf{rcv}\ ackn] \wedge \Diamond_a\,[A\ \mathbf{snd}\ q])$ [by temporal logic]
$\Rightarrow \quad \Diamond\,(p \not\sim q)$ [by axiom A_5 below]
$\Rightarrow \quad p \not\sim q$
q.e.d.

This leads to the extension of the properties part of the specification `CSMA/CD-Implementation` by the following axiom; it claims that the agent A never retransmits a certain frame one it has received an acknowledgment frame, which indicates that the frame has been delivered to B intact.

ENRICH **CSMA/CD-Implementation** BY
PROPERTIES $\forall p, q : \mathit{frame}$
AXM (A_5) : $\blacklozenge_a [A \ \mathbf{snd}\ p] \wedge [A \ \mathbf{rcv}\ \mathit{ackn}] \wedge \blacklozenge_a [A \ \mathbf{snd}\ q] \Rightarrow p \not\sim q$

This completes the refinement of the CSMA/CD protocol.

7 DISCUSSION

In this paper, we have applied a combination of algebraic and temporal techniques for the specification of a CMSA/CD protocol. One of the main points, is that the protocol specification is not presented as a gigantic piece of text, rather is developed step by step starting from a requirement specification that describes the services provided by the protocol. Another point is that in the final product, which is a refinement of the requirement specification, the properties of the individual protocol entities are stated in isolation: an agent occurs only in the formulas that concern its individual behavior. This is very important in the context of distributed systems, since the protocol entities will be implemented in separate locations. Although this protocol is relatively simple, we feel that the methodology could be applied in the development of more complex protocols.

In order to give an easily comprehensible specification we have made some simplifications. We have considered a system with only two stations and unidirectional communication, whereas the protocol is designed to be used with an arbitrary number of stations and bidirectional communication. Moreover, we have investigated an ideal situation, where all agents work perfectly. However, in the real world we have to deal with the possibility of the system failing, e.g. the network itself or the receiver breaking down. This would mean that a frame cannot always be successfully transmitted. In order to model the potential of failure in our specification we have to modify the original *CSMA/CD* specification by replacing the liveness property Prop_2 by the following formula; it states that a frame is either eventually delivered or a failure is reported to the higher layer:

$$[S \ \mathbf{xmt}\ m] \Rightarrow (\Diamond [R \ \mathbf{xmt}\ m] \ \underline{\mathrm{xor}} \ \Diamond [V \ \mathbf{xmt}\ \mathit{fail}(m)])$$

where *fail(m)* is a data element that indicates the cause of failure in transmitting the message m. This can, of course, be specified algebraically.

To conclude, we believe that algebraic-temporal specifications offer a formalism for describing protocols in a sufficiently abstract way. From a practical point of view, they may be used as a formalism for requirement specifications of standard protocols. In this way, they may be applied to verify standard protocols written in LOTOS (Inf 1987), which specifies protocols at a more concrete level.

Acknowledgments: I am grateful to my supervisor, Prof. Peter Pepper, for comments and helpful discussions. Special thanks also to Niamh Warde for correcting the English in this paper.

APPENDIX: SEMANTICS OF THE TEMPORAL LANGUAGE

The underlying model for our temporal language is based on the concept of *events*. An event represents the occurrence of a send, receive, or a transmission action of a message during a system run. The behavior

of a distributed system is considered to be the set of its possible runs, where each run is a sequence of events. The underlying time structure is a linearly ordered set $(\mathcal{E}, \leq)$ of events, in which each event is considered as an atomic point in time. We also assume that $(\mathcal{E}, \leq)$ is infinite and discrete. Since each send or receive action is "equivalent" to a transmission action on the corresponding channel, we may model an event by a pair (S, m), where S is a channel name and m the message.

A *model* $\mathcal{M} = (Alg, (\mathcal{E}, \leq), \sigma)$ for our language consists of an algebra Alg, a time structure $(\mathcal{E}, \leq)$, and an assignment σ that associates a value to each message variable. The algebra Alg specifies a non-empty set of message values and assigns constants and functions to the respective constant and function symbols.

Before giving the semantics of the temporal language we first introduce some abbreviations. Let $(\mathcal{E}, \leq)$ be a linear order, S a channel name and M be a set of channel names.

$$\mathcal{E}_S \stackrel{\text{def}}{=} \{e \in \mathcal{E} \mid \text{there exists a message } m \text{ s.t. } e = (S, m)\}$$

$$\mathcal{E}_M \stackrel{\text{def}}{=} \bigcup_{S \in M} \mathcal{E}_S$$

That is, $\mathcal{E}_S$ denotes the subset of all those events that concern the channel S, and $\mathcal{E}_M$ is the corresponding generalization to sets of channels.

We denote by $\mathcal{M}, e \models P$ the fact that the temporal formula P is valid for the model $\mathcal{M}$ at the point e. This leads to the following semantic definition of the temporal operators:

$\mathcal{M}, e \models \triangle_M P$ iff $e \in \mathcal{E}_M$ and $\mathcal{M}, e \models P$

$\mathcal{M}, e \models \bigcirc_M P$ iff for the least $e' \in \mathcal{E}_M$ with $e < e'$, $\mathcal{M}, e' \models P$

$\mathcal{M}, e \models \Box_M P$ iff for every $e' \in \mathcal{E}_M$ with $e \leq e'$, $\mathcal{M}, e' \models P$

$\mathcal{M}, e \models \Diamond_M P$ iff there exists $e' \in \mathcal{E}_M$ with $e \leq e'$ and $\mathcal{M}, e' \models P$

$\mathcal{M}, e \models \bullet_M P$ iff there exists $e' \in \mathcal{E}_M$ with $e' \leq e$ and $\mathcal{M}, e' \models P$

$\mathcal{M}, e \models \blacklozenge_M P$ iff there exists $e' \in \mathcal{E}_M$ with $e' \leq e$ and $\mathcal{M}, e' \models P$

$\mathcal{M}, e \models P\ {}_M\mathbf{before}_N\ Q$ iff for the least $e' \in \mathcal{E}_N$ with $e < e'$ and $\mathcal{M}, e' \models Q$ there is $e'' \in \mathcal{E}_M$ s.t. $e \leq e'' < e'$ and $\mathcal{M}, e'' \models P$

Atomic formulas: In principle we can use in our logic any predicate on messages, but for our applications all we need are predicates that refer to the activities of transmitting messages. These predicates are defined by

$\mathcal{M}, e \models [S\ \mathbf{xmt}\ m]$ iff $e = (S, \sigma(m))$

$\mathcal{M}, e \models [S\ \mathbf{xmt}\,]$ iff there is $v \in Alg$ s.t. $e = (S, v)$

A temporal formula is called valid in the model $\mathcal{M}$, denoted by $\mathcal{M} \models P$, if $\mathcal{M}, e \models P$ for every $e \in \mathcal{E}$. P is called valid, denoted by $\models P$, if $\mathcal{M} \models P$ for every model $\mathcal{M}$. The validity of a formula P is thus defined by requiring that P holds at all time-points of all models.

REFERENCES

Abrial, J. R. (1985), Programming as a mathematical exercise, *in* C. Hoare, ed., 'Mathematical Logic and Programming Languages', Prentice-Hall International.

Ame (1985), *Local Area Networks: Carrier Sense Multiple Access with Collision Detection (CSMA/CD) Access Method and Physical Layer Specifications.* ANSI/IEEE Standard 802.3.

Barringer, H., Kuiper, R. & Pnueli, A. (1984), Now you may compose temporal specifications, *in* 'Proceding of the 16th ACM Symposium on Theory of Computing', pp. 51–63.

Bjørner, D. & Jones, C. B. (1987), *The vienne development method: the meta-language*, Vol. 280 of *Lecture Notes in Computer Science*, Springer Verlag, Berlin.

Broy, M. (1988), Requirement and design specification for distributed systems, *in* F. Vogt, ed., 'Concurrency', Vol. 335 of *Lecture Notes in Computer Science*, Springer, Berlin, pp. 33–62.

Ehrig, H. & Mahr, B. (1985), *Fundamentals of Algebraic Specification 1*, EATCS Monographs on Theoretical Computer Science, Springer, Berlin.

Hoare, C. (1985), *Communicating Sequential Processes*, Prentice Hall International.

Inf (1987), *LOTOS - A formal description technique based on temporal ordering of observational behaviour.* (ISO/TC 97/SC 21N).

Jmaiel, M. & Pepper, P. (1994), Development of communication protocols using algebraic and temporal specifications, *in* 'Proceedings of the International Workshop on Advanced Software Technology, Shanghai, Sept. 15-16 1994.', Jiao Tong University, Shanghai.

Koymans, R. (1992), *Specifying Message Passing and Time-Critical Systems with Temporal Logic*, Vol. 651 of *Lecture Notes in Computer Science*, Springer, Berlin.

Lichtenstein, O., Pnueli, A. & Zuck, L. (1985), The glory of the past, *in* 'Proc. of the Workshop on Logics of Programs 85', Vol. 193 of *Lecture Notes in Computer Science*, Springer, Berlin, pp. 196–218.

Milner, R. (1980), *A Calculus for Communication Systems*, Vol. 90 of *Lecture Notes in Computer Science*, Springer, Berlin.

Pepper, P. & Wirsing, M. (1994), KORSO: a Methodology for the Development of Correct Software, *in* M. Broy & S. Jähnichen, eds, 'KORSO, Correct Software by Formal Methods', Lecture Notes in Computer Science, Springer.

Petri, C. A. (1962), Fundamantals of a theory of asynchronous information flow, *in* 'Proc. of the IFIP Congress 1962, Munich', North Holland Publishing Company, Amsterdam, pp. 386–390.

Sistla, A., Clarke, E., Francez, N. & Meyer, A. (1984), 'Can message buffers be axiomatized in linear temporal logic?', *Information and Control* **63**, 88–112.

8 BIOGRAPHY

Mohamed Jmaiel received an M.S. degree in computer science from the university of Kiel (Germany) in 1992. In 1992 he applied for a position in the graduate program (Graduiertenkolleg) at the technical university of Berlin. Since then he has been working as a Ph.D. student on a combination of algebraic specifications and temporal logic for the development of communication protocols. His research interests include formal methods, specification and verification of programs, temporal logic and algebraic specifications.