Protocol Specification Using Parameterized Communicating Extended Finite State Machines — A Case Study of the ATM ABR Rate Control Scheme

David Lee
Bell Laboratories
Murray Hill, New Jersey

K. K. Ramakrishnan
AT&T Research
Murray Hill, New Jersey

W. Melody Moh
San Jose State University
San Jose, California

A. Udaya Shankar
University of Maryland
College Park, Maryland

ABSTRACT

Formal specifications are indispensable for computer-aided verification and testing of communication protocols. However, a large number of the practical protocols, including ATM, have only informal specifications, mostly in English. There are no general procedures to derive formal specifications from such informal specifications.

As a case study, we consider an important protocol specification - ATM's Available Bit Rate (ABR) service specification. The ABR source/destination policies have been specified using an English description in the main body of the ATM Forum's draft Traffic Management specification, from which it is hard to conduct a formal analysis. Furthermore, while considerable energy has been spent in providing a reasonably precise specification, while allowing for appropriate implementation latitude, an English description still has the potential for different interpretations.

We model the protocol by parameterized communicating extended finite state machines with timers, which is often called a transitions system, and present a formal specification by transitions of the system. We also provide insights gained in the derivation of the formal specification. Furthermore, we introduce a scheduler, involved in transmitting queued cells at the allowed cell rate to meet the minimal requirements from the source and destination protocols. We present the transitions for the source/destination/scheduler machines, primarily for transmitting cells in-rate.

1. INTRODUCTION

With advanced computer technology, communication protocols are getting larger to fulfill more complicated tasks, however, they are also becoming less reliable. Consequently, verification and testing become an indispensable part of protocol design and implementation; yet it has proved to be a formidable task for complex systems. Formal (automatic) protocol verification and testing has proved to be a powerful tool for the protocol analysis, however, only very few protocol systems have been analyzed formally. A major hurdle is: formal analysis often requires a formal specification of the protocols, yet a large number of the protocols in practice are specified informally, mostly in English.

As a case study, we consider an important ATM protocol - Available Bit Rate (ABR). The ATM Forum's Traffic Management Working Group has been working on a specification for the class of ABR service [ATM]. The service is meant to address the needs of bursty data transfers, and therefore is based on the effective operation of a rate-based congestion control/avoidance mechanism. There are several variants to the possible congestion control algorithms that may be adopted in the network, including a mechanism called the Explicit Forward Congestion Indication (EFCI) scheme [BF95], and a different mechanism called the Explicit Rate (ER) Scheme [CCJ].

The specification for the ABR service is primarily that of the source and destination policies, while the specification of the operation of the intermediate nodes (switches) has been specified somewhat more loosely to allow for vendor latitude. In broad terms, the ABR scheme has sources transmitting data cells at a specified rate, derived using a feedback control mechanism by which the network communicates its state. Periodically (once every *Nrm* data cells) sources transmit a resource management cell (RM cell). This RM cell serves the function of a probe into the network that is used to detect the state of the network (congested or otherwise). The RM cell also serves as a cell in which the network may communicate an explicit rate back to the source to indicate the rate at which the source is permitted to transmit. RM cells sent by a source are received by the remote destination and have to be turned around and sent back. This function is performed by the destination portion of the ABR specification.

The source and destination policies have been specified using an English description in the main body of the draft Traffic Management specification [ATM]. Given a protocol of such complexity, it is out of question to perform manually a formal analysis, such as assertion proofs. It is also impossible to conduct an automatic verification/testing, since computer-aided techniques require a formal specification and there are no general procedures to obtain such specification from the English description. Furthermore, while considerable energy has been spent in providing a reasonably precise specification, an English description may sometimes lead to implementations that do not meet the letter and/or the spirit of the specification.

We use a Parameterized Extended Finite State Machines (PEFSM) with Timers to model and specify the source/destination behavior. As in a Finite State Machine, there are states and transitions between states. In addition, we have variables and each transition is associated with a predicate (event) and an action; a transition is executable if the associated predicate is TRUE with the current variable values and in this case the action updates the variable values and the machine moves to the next state. On the other hand, there are parameters associated with the input/output cells (messages) to/from the machines and they affect the execution of the transitions and the variable values. Furthermore, there are timers and the machine behavior also depends on the timer values. Such a

0-8186-7453-9/96 $05.00 © 1996 IEEE

representation often is also called a transition system. Communicating and extended finite state machines have been used to model protocols [LY]. To properly model the parameters in the ABR protocol associated with the input/output (RM and data cells), we need another level of generalization of Parameterized EFSM's. For the ABR protocol, we have three machines: Source, Destination, and Scheduler, which correspond to the Source, Destination, and Scheduler behaviors, respectively. For each machine, we have a set of states, variables, and a list of transitions between states.

There are several subtleties that the PEFSM captures that are likely to be subject to varying interpretations when reading the English description. For example the inter-operation of the EFCI scheme and the ER scheme may be precisely captured with a PEFSM model, as we have attempted to perform here.

Implied in the English specification is the need for a scheduler at the source which paces out data and RM cells. Data cells are transmitted at a rate of *ACR*, where *ACR* is the currently Allowed Cell Rate. After every $Nrm - 1$ data cells, a Forward RM (FRM) cell is transmitted by the source, maintaining the same *ACR* rate. FRM cells received from a remote station are turned-around and transmitted back as "turned-around" backward RM (BRM) cells. Furthermore, a fairly stringent rule has been specified for the ordering of the transmission of the different types of cells from the source: data cells vs. FRM cells vs. turned-around Backward RM (BRM) cells. We identify in this paper the minimal requirements for the scheduler to match the specification. The protocol specification requires the source to transmit FRM cells based on a timer. Moreover, when the time between transmission of successive forward RM cells is excessive, a rate change is specified in the protocol. All of these require an interface between the source protocol state machine and the scheduler that reflects the specification. We identify the minimal interface between the source, destination and scheduler machines, sufficient to reflect the specification accurately.

The paper is organized as follows. Section 2 provides an overview of the interdependencies between the various protocol state discuss the organization of the state machines. We present the formal specifications for the source, destination, and scheduler machines in the following three sections, respectively. In each section, we illustrate the transitions of the PEFSM, comparing them with the corresponding rules in the original specification [ATM].

2. ORGANIZATION OF THE STATE MACHINES

Since the source and destination protocols for the ABR Traffic management scheme are to achieve a closed-loop feedback control system, and furthermore the Virtual Circuit (VC) is full-duplex, each of the protocols involves both sending information as well as receiving feedback from the remote system. We show in Figure 1, the relationship between the state machines at both the sending (called station A) and receiving (called station B) stations. The sending station has a source protocol machine which is involved in initialization, and sending data cells and RM cells to station B via a scheduler submachine.

We break up the source machine into 2 parts.

(a) A source submachine that models all of the non-timing related functions of changing the rate, and servicing BRM cells queued by the destination. These BRM cells are then queued to the second submachine.

(b) The second submachine is called the scheduler, which performs all the timing related functions of the source specification. Specifically, source Rules 3 and 5 are incorporated in the scheduler submachine.

The interface between the source protocol submachine and the scheduler submachine has the following three queues: (i) A queue of turned-around BRM cells that have to be sent back to station B. This queue is between the source and the scheduler. (ii) Two "virtual queues" of data cells and FRM cells. When the scheduler submachine determines it is time to send an FRM cell (e.g., station A generates an FRM cell every $Nrm - 1$ data cells), it creates an FRM (as if picking up an FRM waiting in its virtual queue), fills in the relevant information in its fields and transmits it. The "virtual queue" for data cells is again used by the scheduler submachine to pick up a data cell from the user to transmit when it is time to send one.

The scheduler submachine keeps track of time "T", which is the time since the transmission of the last FRM cell, as also the time "t" between individual cells of the VC, which are separated by time $(1/ACR)$.

An FRM cell received at a station (e.g., station B) comes into the destination protocol state machine, and is converted into a "turned-around" BRM cell. This turned-around BRM cell has to be transmitted back to the other station (in Figure 1, back to station A). We go through an interface between the destination protocol machine and the source protocol machine at station B, through a queue that only contains turned-around BRM cells. Then the source hands the turned-around BRM cell to the scheduler state machine (scheduler at station B), which transmits it on the link. The cell is now delivered as a BRM cell to the source protocol machine at station A. The source protocol machine receives this BRM cell and takes the appropriate action (typically, alter the source transmission rate, *ACR*).

We therefore have the destination protocol state machine interface with the source through a queue (for turned-around BRM cells). The source protocol state machine then interfaces with the scheduler state machine through 3 (for data, turned-around BRM cells and FRM cells respectively) queues. The scheduler finally transmits the appropriate cells based on the protocol specification.

Each machine has several states, variables and transitions between states. A transition is executable if the associated event is TRUE based on the current variable values and in this case the action updates the variable values. There is a number of parameters [ATM]: BN, CI, DIR, EFCI, ER, and NI. They are stored/modified in the input/output cells and they affect the events and actions of the transitions. There are also system parameters [ATM]: ADTF, ICR, MCR, Mrm, Nrm, PCR, RDF, RIF, and Trm. These constant parameters (for a particular system) remain unchanged during the execution of the transitions and are not associated with the data and RM cells.

3. SOURCE MACHINE (SOURCE BEHAVIOR)

The main parts of the source behavior modeled here is the actions that change the rate upon the arrival of a BRM cell returned from the remote destination, and providing the interface between the destination protocol machine and the scheduler to transmit turned-around BRM cells. The interface with the destination protocol machine is through a queue of BRM cells queued to the source.

We are primarily concerned with modeling the actions that are specified when transmitting cells "in-rate". This means that any time a cell is ready for transmission, it is allowed to be transmitted only after a minimum time of ($1/ACR$) has elapsed since the transmission of the previous cell from this VC. Data as well as RM cells may be transmitted "out-of-rate", when a cell opportunity has not yet arrived for this VC. However, these issues implementation dependent, which we have elected to ignore.

The source machine has a total of 4 states. The primary state S_2 is involved in changes to the rate when a BRM cell is received (the other states when the rate is altered when an FRM cell is transmitted are incorporated in the scheduler submachine). A local count of the number of FRM cells and turned-around BRM cells queued to the scheduler are shared variables with the scheduler. Similarly, there is a shared variable for the number in the queue between the destination and source protocol machines. Operations to the shared variables need to be coordinated between the source protocol machine and the other corresponding machine. We indicate this by requiring a lock to be acquired for the variable prior to updating it.

The other event that the source machine deals with is the arrival of a turned-around BRM cell from the destination machine that has to be transmitted.

The initial transition, T_1, occurs when the connection is set up and the first data cell is available for transmission (Source Rule 2). The first action performed is to initialize the rate to *ICR* and move to an active state. When a BRM cell arrives, transition T_2 or T_3 is fired, leading to one of the transitions T_4 through T_7 where a rate change (modification of *ACR*) takes place.

The interface between the source EFSM and the destination EFSM is typical of many bi-directional protocols. There is a need for the protocol machine that receives information to turn-around and transmit some corresponding information (e.g., acknowledgements). In the ABR case, the destination protocol machine turns around FRM cells received as "turned-around BRM cells". These have to be merged into the stream of cells sent in the forward direction, the data and FRM cells. We seek a minimal interface between the source protocol machine and the destination protocol machine. Instead of keeping track of individual RM cells queued by the destination protocol machine to the source, we maintain a simple counter of the number of turned-around BRM cells queued to the source by the destination machine, and a single copy of the contents of the RM cell queued. This is because we are able to exploit the nature of the interface, which is that all the turned-around BRM cells queued at the source are either dropped or updated with the latest RM cell information received by the destination. This allows us to retain the semantics of an EFSM for each of the state machines, without having to keep a queue of all of the individual RM cells queued, and having to search through the queue when altering these RM cells, as required by the destination rules.

The interface between the destination protocol machine and the source protocol machine is used in transition T_8, where we determine if a turned around backward RM cell queued to the source for transmission has to be prepared for the scheduler. Again, when the turned-around backward RM cell is queued to the scheduler by the source, transition T_9, we follow similar rules. If there are no turned-around backward RM cells in the queue, we simply perform the queueing operation for subsequent transmission. If there are turned-around BRM cells in the queue, then we reflect once again the rules specified in the English specification (i.e., to drop all the previously queued turned-around BRM cells or to modify all of them) in the destination policy. We do so by non-deterministically choosing between transition T_{10} and T_{11}, to reflect the fact that this is implementation dependent.

When a BRM cell is received by the source, this is the return of feedback information from the network which is to be acted upon by the source. The action is primarily to modify the transmission rate, ACR, by the source. Transitions T_2 and T_3 recognize the receipt of the backward RM cell from the remote end, and capture the information returned in the RM cell when the source gets feedback from the switches (and possibly the destination) operating in the EFCI mode. We distinguish by BN_{BRM} those BRM cells received that have made the entire round-trip as opposed to those BRM cells that were generated from the network in a Backward Explicit Congestion Notification mode. Transitions T_4 and T_7 reflect the manner in which the source modifies the ACR, when it is responding to switches operating in both the EFCI mode as well as the ER mode. The CI field in the returned RM cell takes effect first to reduce the rate if any, in transition T_4, and then the Explicit Rate ER_BRM returned in the BRM cell is used to reduce the source's ACR subsequently. Similarly, increases to ACR occur only if the switches/destination operating in the EFCI mode do not feedback that they are congested, and the explicit rate returned from ER (Explicit Rate) switches allows ACR to be increased from its current value.

The transition diagram of the Source Machine is in Figure 2.

SOURCE MACHINE

STATES

S_0: initial state: connection set up and first data cell available

S_1: active state

S_2: state in which rate may be changed

S_3: state in which queued BRM cells are altered

VARIABLES

N: number of turned-around BRM cells queued to Source by Destination /* shared variable with Destination */

X: number of BRM cells queued to Scheduler /* shared variable with Scheduler */

Y: number of FRM cells sent by Scheduler since last BRM cell with BN=0 received by Source /* shared variable with Scheduler */

EVENTS

E_1: BRM cell received by Source

E_2: data cell waiting to be sent

TRANSITIONS

T_1 (Source 2)
current state: S_0
next state: S_1
event:
connection set up complete and E_2
actions:
$ACR :\leq ICR$;
$N := 0$;
$X := 0$;

T_2 (Source 8)
current state: S_1
next state: S_2
event:
E_1 $(BN_{BRM} = 0)$
actions:
$CI := CI_{BRM}$;
$NI := NI_{BRM}$;
$Y := 0$; /* lock Y */

T_3 (Source 8)
current state: S_1
next state: S_2
event:
E_1 $(BN_{BRM} = 1)$
actions:
$CI := CI_{BRM}$;
$NI := NI_{BRM}$;

T_4 (Source 8 and 9)
current state: S_2
next state: S_1
event:
$CI = 1$
actions:
$ACR := \max\{MCR, \min\{ACR*(1 - RDF), ER_{BRM}\}\}$;

T_5 (Source 8 and 9)
current state: S_2
next state: S_1
event:
$((CI = 0)\ \&\&\ (NI = 0))\ \&\&\ (ACR < ER_{BRM})$
actions:
$ACR := \max\{MCR, \min\{ER_{BRM}, PCR, ACR + RIF*PCR\}\}$;

T_6 (Source 8 and 9)
current state: S_2
next state: S_1
event:
$((CI = 0)\ \&\&\ (NI = 1))\ \&\&\ (ACR < ER_{BRM})$
/* $NI = 1$ prevents a rate increase */
actions:

T_7 (Source 8 and 9)
current state: S_2
next state: S_1
event:
$(CI = 0)\ \&\&\ (ACR \geq ER_{BRM})$
actions:
$ACR := \max\{MCR, ER_{BRM}\}$;

T_8 (Source implied)
current state: S_1
next state: S_3
event:
$N > 0$
actions:

T_9 (Destination 3; Source queues BRM cell)
current state: S_3
next state: S_1
event:
$(X = 0)$ /* Scheduler queue has no BRM cells */
actions:
queue this BRM cell to Scheduler;
$N := N - 1$; /* lock N and X */
$X := 1$;

T_{10} (Source queues BRM cell for BRM dropping)
current state: S_3
next state: S_1
event:
$(X > 0)\ \&\&\ (BN_{BRM} = 0)$ /* Scheduler queue has BRM cells */
actions:
$N := N - 1$; /* lock N */
drop all queued BRM cells at Scheduler queue;
queue this BRM cell to Scheduler;
$X := 1$; /* lock X */

T_{11} (Source queues BRM cell for BRM rewriting)
current state: S_3
next state: S_1
event:
$(X > 0)\ \&\&\ (BN_{BRM} = 0)$ /* Scheduler

queue has BRM cells */

actions:

$N := N - 1$; /* lock N */

rewrite all queued BRM cells at Scheduler queue;

queue this BRM cell to Scheduler;

$X := X + 1$; /* lock X */

T_{12} (Source queues BRM cell with $BN_{BRM} = 1$)

current state: S_3

next state: S_1

event:

$(X > 0)$ && $(BN_{BRM} = 1)$ /* Scheduler queue has BRM cells */

actions:

$N := N - 1$; /* lock N */

queue this BRM cell to Scheduler; /* Destination turned-around BRM cell with BN=1 does not alter the queued BRM cells in Scheduler */

$X := X + 1$; /* lock X */

4. DESTINATION MACHINE (DESTINATION BEHAVIOR)

The destination protocol machine is somewhat simpler than the source and scheduler machines, even though there are almost the same number of transitions. This is primarily to reflect all the variants of EFCI mode and ER mode operations of the network that are contained in the ABR service specification, when the destination gets an FRM cell and has to prepare it to send back as a turned-around BRM cell.

There are only two states involved in the destination machine. The shared variable N is the number of turned-around BRM cells queued by the destination protocol machine to be transmitted by the source.

Figure 1 shows the interaction of the source, destination, and scheduler machines. FRM cells arrive at the destination, the destination turns them around and queues them (they now become turned-around BRM cells) to the source, and the source queues the turned-around BRM cells to the scheduler to be sent to the network

The primary event that the destination machine operates from is the arrival of an FRM cell at the destination. This causes the destination to turn around the cell as a BRM cell in the destination rule 2 of the English specification. We reflect this in the transitions T_2 through T_8. The alternatives of whether to set the No Increase (NI) bit is implementation specific. The events for the corresponding transitions may not be mutually exclusive, resulting in non-determinism in the EFSM model. In each case, the appropriate fields of the BRM cells are set as specified in the corresponding English rule, and we move to a state where the BRM cells are queued to the source.

Transition T_1 reflects the destination rule 1 which specifies the accumulation of EFCI information received from the network in data cells. We collect the EFCI bit received in the data cells in the local variable *saved_*EFCI at the destination.

Transitions T_9 and T_{10} reflect destination rule 6 of the English specification which allows the destination to generate a BRM cell exogenously when it is congested. Since it is implementation dependent, as to whether the NI bit is set or not, the choice to follow transitions T_9 or T_{10} is again non-deterministic.

Transition T_{11} finally hands the BRM cell to the source machine, for it to "transmit" (i.e., hand to the scheduler). The interface between the destination machine and the source machine, is a minimal one. The only shared variable is the counter N, which is incremented in transition T_{11}, when a turned-around BRM cell is queued. Another alternative to transition T_{11} is transition T_{12} where all the queued turned-around BRM cells are dropped. Once again, the choice between T_{11} and T_{12} is implementation dependent, and therefore non-deterministic in our EFSM model. The destination rule allows for either all of the queued turned-around BRM cells to be dropped, or updated with the latest information received in the current turned-around BRM cell. When the queued turned-around BRM cells are dropped, the counter for the number in the queue, N is set to 1, in transition T_{12}.

Note that the alteration of the queued turned-around BRM cells is only of those cells queued at the interface with the source protocol machine. There is another queue of such cells between the source and the scheduler, which we also look at; this is described in the scheduler machine in the next section.

The transition diagram of the Destination machine is in Figure 3.

DESTINATION MACHINE

STATES

S_0: wait for cells

S_1: state in which BRM cells are queued to Source and source queue of BRM cells may be altered

VARIABLES

saved_EFCI: EFCI state of connection saved at Destination

N: counter of the number of BRM cells turned around by Destination machine and queued to Source /* shared variable with Source */

EVENTS

E_1: an FRM cell received

E_2: a data cell received

E_3: destination has internal congestion

TRANSITIONS

T_1 (Destination 1)

current state: S_0

next state: S_0

event:

$E_2(EFCI_{DATA})$

actions:

$saved\_EFCI := EFCI_{DATA}$;

T_2 (Destination 2a)
current state: S_0
next state: S_1
event:
E_1 && (*saved_EFCI* = 0) && (!E_3)
actions:
DIR := 1;
BN := 0;
CI := 0;
NI := 0;
saved_EFCI := 0;

T_3 (Destination 2a)
current state: S_0
next state: S_1
event:
E_1 && (*saved_EFCI* = 1) && (!E_3)
actions:
DIR := 1;
BN := 0;
CI := 1;
saved_EFCI := 0;

T_4 (Destination 2b, for ER Destination)
current state: S_0
next state: S_1
event:
E_1 && E_3 && (*saved_EFCI* = 0)
actions:
DIR := 1;
BN := 0;
CI := 0;
NI := 0;
saved_EFCI := 0;
reduce *ER*;

T_5 (Destination 2b, for non-ER Destination)
current state: S_0
next state: S_1
event:
E_1 && E_3 && (*saved_EFCI* = 0)
actions:
DIR := 1;
BN := 0;
CI := 0;
NI := 1;
saved_EFCI := 0;

T_6 (Destination 2b, for non-ER Destination)
current state: S_0
next state: S_1
event:
E_1 && E_3 && (*saved_EFCI* = 0)
actions:
DIR := 1;
BN := 0;
CI := 1;
saved_EFCI := 0;

T_7 (Destination 2b, for ER Destination)
current state: S_0
next state: S_1
event:
E_1 && E_3 && (*saved_EFCI* = 1)
actions:
DIR := 1;
BN := 0;
CI := 1;
saved_EFCI := 0;
reduce *ER*;

T_8 (Destination 2b, for non-ER Destination)
current state: S_0
next state: S_1
event:
E_1 && E_3 && (*saved_EFCI* = 1)
actions:
DIR := 1;
BN := 0;
CI := 1;
saved_EFCI := 0;

T_9 (Destination 5, cause rate reduction)
current state: S_0
next state: S_1
event:
E_3
actions: /* Destination generates BRM cell */
DIR := 1;
BN := 1;
CI := 1;

T_{10} (Destination 5, prevent rate increase)
current state: S_0
next state: S_1
event:
E_3
actions: /* Destination generates BRM cell */
DIR := 1;
BN := 1;
CI := 0;
NI := 1;

T_{11} (Destination implicit for BRM rewriting)
current state: S_1
next state: S_0

event:

$(N > 0)$ && $(BN = 0)$

actions:

rewrite all queued BRM cells at Source with the contents of this BRM cell;
queue this BRM cell to Source;
$N := N + 1$; /* lock N */

T_{12} (Destination implicit for BRM dropping)

current state: S_1
next state: S_0
event:

$(N > 0)$ && $(BN = 0)$

actions:

drop all queued BRM cells at Source;
queue this BRM cell to Source;
$N := 1$; /* lock N */

T_{13} (Destination implicit for BRM cell with $BN_{BRM} = 1$)

current state: S_1
next state: S_0
event:

$(N > 0)$ && $(BN = 1)$

actions:

queue this BRM cell to Source; /* Destination turned-around BRM cell with BN=1 does not alter the queued BRM cells at Source */
$N := N + 1$; /* lock N */

T_{14} (Destination implicit)

current state: S_1
next state: S_0
event:

$N = 0$ /* queue is empty */

actions:

$N := 1$; /* lock N */
queue this BRM cell to Source;

5. SCHEDULER MACHINE (SCHEDULER BEHAVIOR)

This is the primary section of the EFSM model of the entire source/destination behavior where we have had to use our understanding and interpretation of the English specification to arrive at the minimalistic representation adequate to have "reasonably" correct behavior of the ABR service. The scheduler is a machine that is intentionally left to implementation by vendors according to their needs and expertise, to allow for vendor differentiation. However, there is a minimal requirement of the scheduler, which is that cells from the VC (RM cells as well as data cells) are not transmitted at a greater rate than *ACR*. By this we mean that the time between successive transmission of cells for a VC is no less than $(1/ACR)$. We have ensured that this is reflected in the EFSM, while not necessarily incorporating all of the implementation issues relating to when there are multiple VCs awaiting transmission at the source on a cell time etc., which we feel is outside the essential scope of this formal specification.

The source policy for ABR specifies the time at which different types of cells (FRM, turned-around BRM and data cells) are to be transmitted. FRM cells are transmitted after every $Nrm - 1$ data and BRM cells have been transmitted. In addition, FRM cells are transmitted upon a timer expiration, so as to ensure that a "probe" for feedback information is transmitted with a minimum frequency. At the time an FRM cell is transmitted based on the timer expiration, actions specified by the source policy modify the allowed rate, as well as other variables. To avoid having a complex interface between the source protocol state machine and the scheduler machine, we have partitioned the state transitions relating to the source policy between the two machines. Thus, the number of shared variables between the state machines is kept small (only relating to the state of each of the queues). In fact, the queue of data cells is entirely within the purview of the scheduler machine alone, since all of the actions that relate to data cell transmissions are represented in the transitions within the scheduler.

The scheduler machine keeps track of time "*T*", which is the time since the transmission of the last FRM cell, as also the time "*t*" between individual cells of the VC, which are separated by time $(1/ACR)$. However, these variables are local to the scheduler machine.

The shared variable ACR between the scheduler and the source, the counter for the size of the BRM cells queued and the number of FRM cells sent by the scheduler since the last BRM cell was received by the source (feedback from the network) are shared variables that have to be locked for update by the 2 machines. But, since the source updates the shared variable (i.e., *ACR*) in zero time, the impact of the locking by the source is minimal - the scheduler is delayed only by time 0, when it needs access to the variable *ACR* or queue size. This allows us to have both submachines have "concurrent" access to the variables with no performance impact.

The timer, *T*, is used to track the amount of time since the last forward RM cell was transmitted by the scheduler. When there are multiple streams (multiple source machines) maintaining per-stream timers it is important that the timer reflect the time instant from when the data was transmitted from the scheduler rather than when the data was handed by the source machine. Thus, local congestion at the sending station may be handled appropriately, and the timer would reflect the true value being measured. The scheduler keeps track of the number of turned-around BRM cells in the queue to be transmitted in the shared variable *X*, and the number of FRM cells transmitted by the scheduler since the last FRM cell was received, in the shared variable *Y*. The local time maintained by the scheduler, to ensure that successive cells for the VC are sent after at least $(1/ACR)$ is a local timer *t*. When it is time to transmit an FRM cell, we model the source behavior as the scheduler creating an FRM cell with the appropriate fields and transmitting it. Thus, FRM cells are not actually queued between the source and the scheduler. The queue of data cells is a virtual one, since we do not model the higher layer handing data cells to be sent. The

scheduler only recognizes the event, E, when a data cell is available to be transmitted.

Because of the way we model the interfaces between the destination-source-scheduler, we ensure that the turned-around BRM cell arrives via source machine. We believe that this is the correct way to describe what typically happens in an end-system when there is received information that has to be turned-around and transmitted. It is best to interface this via the source protocol machine and keep the interfaces simple.

Transitions T_2 and T_3 combine the conditions specified by both rules 3a and 5, that relate to the transmission of an FRM cell. The predicates for transition T_2 specify when an FRM cell may be transmitted. If source rules 3a and/or 5 are enabled, i.e., transition T_3's predicate is satisfied, we move to a state S_2, and the current rate, ACR, is set to the initial rate, ICR, as specified by source rule 5.

In transition T_5, a rate change may occur at the time an FRM cell has to be transmitted, reflecting the actions specified by source rule 6. This involves a test of the counter Y (number of FRM cells queued to the scheduler since the last BRM cell with $BN = 0$ was received), and implementing a rate reduction if the counter has exceeded a threshold.

If we think of a cycle of transmission as being an FRM cell, one or more turned around BRM cells and $Nrm - 2$ data cells, the source policy provides a dynamic priority for the transmission of FRM cells and data cells within the cycle. Subsequent to transmission of the FRM cell at the beginning of this "imaginary cycle", the source policy provides priority to transmit a queued turned-around BRM cell, prior to transmission of data cells. Subsequent to the transmission of the turned-around BRM cell, data cells have priority, until $Nrm - 2$ data cells have been transmitted. This broad principle, as specified in rules 3(b) and 3(c) in the source policy, is reflected in transitions T_6 and T_7 in our model here.

Transitions T_2, T_6, and T_7 transmit the appropriate cell when the local timer t has exceeded the minimum time ($1/ACR$), and there is something to be sent. The corresponding counters are modified, and t reset.

The transition diagram of the Scheduler machine is in Figure 4.

SCHEDULER MACHINE

STATES

S_0: initial state

S_1: active state

S_2: state in which rate may be changed

VARIABLES

X_1: number of BRM cells transmitted by Scheduler since last FRM cell transmitted by Scheduler

X_2: number of data cells transmitted by Scheduler since last FRM cell transmitted by Scheduler

X: number of BRM cells queued to Scheduler by Source /* shared variable with Source */

Y: number of FRM cells sent by Scheduler since last BRM cell with BN=0 received by Source /* shared variable with Source */

TIMER

t: time elapsed since last cell sent by Scheduler

T: time elapsed since last FRM cell sent by Scheduler

EVENTS

E: data cells waiting to be sent

TRANSITIONS

T_1 (initial set up)

current state: S_0

next state: S_1

event:

actions:

$X_1 := 0;$

$X_2 := 0;$

$Y := 0;$

$T := 0;$

$t := 0;$

T_2 (Source 3a and 5)

current state: S_1

next state: S_1

event:

$(((X_1 + X_2 \geq Mrm)\ \&\&\ (T \geq Trm))\ \|$
$(X_1 + X_2 = Nrm - 1))\ \&\&$
$((ACR \leq ICR)\ \|\ (T \leq ADTF))\ \&\&\ (t \geq 1/ACR)$

actions:

$CCR_{FRM} := ACR;$

send an FRM cell;

$Y := Y + 1;$ /* lock Y */

$X_1 = 0;$

$X_2 := 0;$

$t := 0;$

$T := 0;$

T_3 (Source 3a and 5)

current state: S_1

next state: S_2

event:

$(((X_1 + X_2 \geq Mrm)\ \&\&\ (T \geq Trm))\ \|$
$(X_1 + X_2 = Nrm - 1))\ \&\&$
$(ACR > ICR)\ \&\&\ (T > ADTF)$

actions:

$ACR := ICR;$ /* lock ACR */

T_4 (Source 3a, 5, and 6)

current state: S_2

next state: S_1

event:

$Y < Crm$

actions:

$CCR_{FRM} := ACR$;
send an FRM cell;
$Y := Y + 1$; /* lock Y */
$X_1 := 0$;
$X_2 := 0$;
$t := 0$;
$T := 0$;

T_5 (Source 6 in the context of Source 5 and 7)

current state: S_2
next state: S_1
event:

$Y \geq Crm$

actions:

$ACR := \max\{ MCR, ACR*(1 - CDF)\}$; /* lock ACR */
$CCR_{FRM} := ACR$;
send an FRM cell;
$Y := Y + 1$; /* lock Y */
$X_1 = 0$;
$X_2 := 0$;
$t := 0$;
$T := 0$;

T_6 (Source 3(b) send a BRM cell to network)

current state: S_1
next state: S_1
event:

$(X > 0)$ && $((X_1 + X_2 < Mrm)$ || $(T < Trm))$ && $(X_1 + X_2 < Nrm - 1)$ && $((X_1 = 0)$ || $!E)$ && $(t \geq 1/ACR)$

actions:

$X := X - 1$; /* lock X */
$X_1 := X_1 + 1$;
send a BRM cell;
$t := 0$;

T_7 (Source 3(c) send a data cell to network)

current state: S_1
next state: S_1
event:

E && $((X = 0)$ || $(X_1 > 0))$ && $((X_1 + X_2 < Mrm)$ || $(T < Trm))$ && $(X_1 + X_2 < Nrm - 1)$ && $(t \geq 1/ACR)$

actions:

$X_2 := X_2 + 1$;
send a cell (data cell);
$t := 0$;

6. CONCLUSION

In this paper, we have described a parameterized communicating extended finite state machines model for the ABR service specification currently being considered in the ATM Forum. While it is often appropriate to develop a congestion control mechanism and the protocol through English descriptions and simulations etc., it is valuable to then derive a formal model based on that description. This allows for an unambiguous interpretation of the specification by implementors to allow for interoperability between implementations. The formal specification would also facilitate formal analysis, such as verification and testing, of the ABR protocol. Based on the formal specification, preliminary results have been obtained on the correctness and performance of the ABR protocol [LRMS97].

We believe that the formal model for the source, destination, and the scheduler developed here clarify what appears on the surface to be a complex English specification. We have introduced the concept of a scheduler (that was only implicit in the English specification), and two simple interfaces between the source, destination and the scheduler machines. Using these interfaces, and interpreting the English description of the source and destination rules, three corresponding PEFSM's have been developed. The novelty is in the interfaces we have introduced among the three machines, and the recognition that the destination protocol machine interfaces to the source protocol machine only. Finally, the scheduler interfaces only to the source protocol machine.

ACKNOWLEDGEMENT

Valuable comments from John Kenney and Huayan Wang in the initial stages of this work are deeply appreciated.

REFERENCES

[ATM]ATM Forum Traffic Management Specification Version 4.0, ATM Forum/95-0013R11, March 1996.

[BF95]Bonomi, F. and Fendick, K. W., "The Rate-Based Flow Control Framework for the Available Bit Rate ATM Service", IEEE Network, Vol. 9, No. 2, March/April 1995.

[CCJ] A. Charny, D. D. Clark, and R. Jain, "Congestion Control with Explicit Rate Indication," *Proc. ICC*, June 1995.

[LRMS]David Lee, K. K. Ramakrishnan, W. Melody Moh and A. Udaya Shankar, "Performance and Correctness of the ATM ABR Rate Control Scheme," submitted for publication.

[LY] D. Lee and M. Yannakakis, "Principles and Methods of Testing Finite State Machines - a Survey," to appear in *The Proceedings of IEEE.*

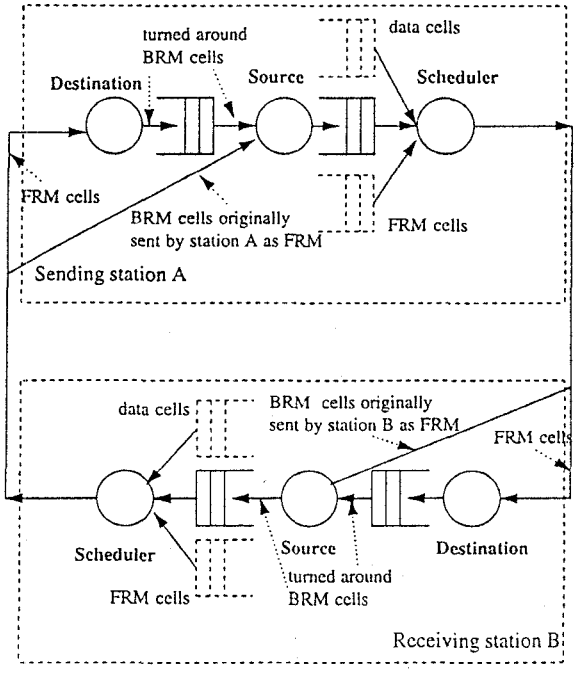

Figure 1: Overview of the protocol state machines and cell flow between source and destination stations

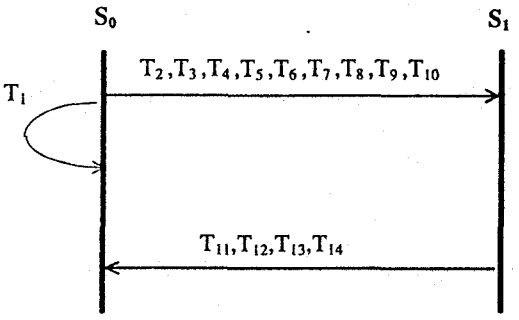

Figure 3: Destination machine

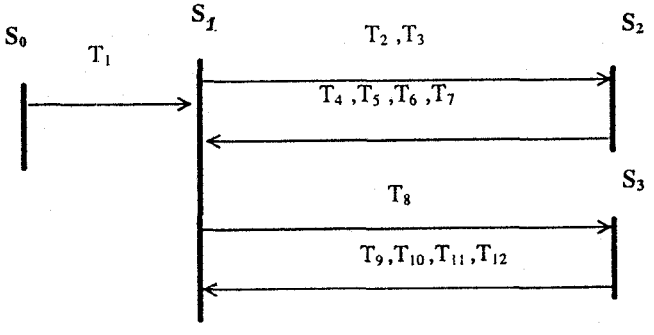

Figure 2: Source machine

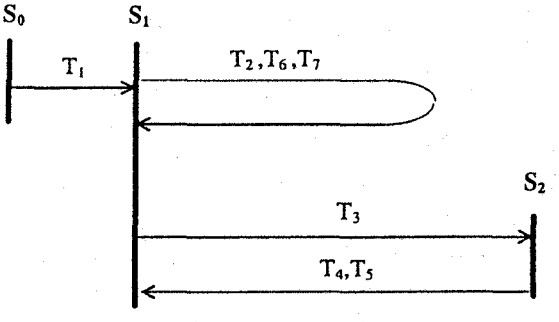

Figure 4: Scheduler machine