Preventing Rather Repairing - A New Approach in ATM Network Management

Anja Schuhknecht <schuhknecht@lrz-muenchen.de>
Gabi Dreo <dreo@lrz-muenchen.de>

Abstract

The introduction of new network services, broadband multi-media and advanced mobile communications imposes high requirements on ATM network management. In such an environment, an essential task of ATM management is to prevent faults and to support efficient resource allocation. The idea is to specify parameters, according to which the behaviour of the network is monitored, and to perform the necessary actions with respect to history-based traffic information before a degradation of quality of service is recognized by users. By describing a model, including parameters, actions, policies, and control mechanisms, we present a step towards self-regulating network management.

1 Introduction

To cope with the requirements that ATM technology imposes on network management, it is necessary to discuss the changing role of network management. Currently, network management is concerned mainly with the support of isolating faults and detecting performance bottlenecks by long-term monitoring. However, having in mind the possibility to control the traffic in the network, the goal should be to **prevent** faults (i.e., unacceptable high cell losses), and to support efficient resource allocation in the ATM network.

The basic principle of ATM is statistical multiplexing, the success of which depends to a great extent on the statistical traffic behaviour in the network. It is therefore necessary to adjust resource allocation schemes with respect to certain characteristics of the traffic. Traditionally, network management is used to monitor the network, and the operator adjusts the network configuration after problems were detected.

A desired goal, however, is self-regulating network management, which should be able to configure the appropriate parts of the network with respect to the special requirements of the users **before** they recognize problems. These configuration decisions should be based on monitored traffic and knowledge of history network behaviour.

The necessity of a new role of ATM management has been recognized by various other authors. [8] introduces an architecture for the integration of performance management and resource control. Thereby providing the network operator with a set of control parameters to actively influence the way the Connection Admission Control (CAC) operates. The architecture they present is based on previous work of [5] and [6], and provides the possibility to adjust the CAC according to performance requirements. However, the choice of reasonable values for the control parameters is based on the experiences of the operator and the system acts on previously detected call request intensities. Unlike the approach presented in this paper, they do not consider any knowledge about *usual* (i.e., history-based) traffic requirements.

In another work of the same group, [1] presents an algorithm for virtual path configuration with the goal of minimizing call blocking and call setup times. So, this work aims in the prevention of problems, as well.

[4] also notes the necessity for management platforms to "quickly determine and execute corrective actions" in response to new technologies and services. They propose an intelligent multi-agent system where agents can work together and exchange knowledge about the *current* network state. As an example application of this system an 'early-cell-discard' approach is presented. There, the aim is to discard cells as early as possible, when congestion appearing further on the network would cause them to get lost anyway. On the contrary, the approach proposed in the present paper tries to minimize the occurrence of congestion itself.

[11] deals with the complex aspects of history management in general. This work presents an architecture to support the operator in the prediction of future behaviour of the network by providing him with a suitable environment "to analyze the conditions leading to undesirable states" in the past. However, the importance of combining history information and resource allocation objectives, especially in ATM networks, has not been considered up to now.

While recognizing the importance and necessity of the above approaches the work presented here takes a different route. The motivation is quite clear: As the success of a generous bandwidth sharing strategy in ATM depends on the statistical behaviour of the traffic - which certainly varies depending on providers, customers, available services and time - the base for management operations should be build on knowledge about this traffic.

In order to develop an adaptive algorithm, it is necessary to derive definite operations, that regulate the traffic in our network in order to match all user requirements best.

This work tries to identify such operations and the correct time to initiate them. A model is presented, which allows the derivation of management parameters to monitor and the actions to be carried out when certain values are exceeded in order to achieve the overall goal of 'Preventing Rather Repairing'.

To sketch the tasks to solve, it is necessary (i) to specify parameters to monitor the behaviour of the network, (ii) to specify thresholds for triggering the necessary actions, and (iii) of course, to specify the actions themselves. We present a model where these issues are tackled in order to achieve the goal of a self-regulating network management.

The paper proceeds as follows: Section 2 discusses network management tasks and introduces the 'connection-oriented view' for the definition of management goals. Afterwards, resource allocation is analyzed in section 3. Section 4 describes the proposed model and its main components: Specified parameters, actions, policies, and control mechanisms. In section 5, we discuss the implementational experiences gained. Section 6 concludes the paper and presents an outlook on future research.

2 Tasks of Network Management

As already stated above, network management systems are usually used to monitor and log the behaviour of the network components with the goal of detecting problems before the user does. Therefore, currently the main objective of network management is to provide the network operator with the capability to **react** as fast as possible to problems in the network.

Considering network technologies used so far, this is a quite reasonable role for network management. If a high collision rate on an Ethernet segment is detected, the required action of the network operator would be to split the problematic segment. There are no possibilities for the NMS to **act** instead of the operator. This view of Network Management Systems can be called a 'device-oriented view', as the focus is laid on monitoring the *correct* and best operation of the device.

With ATM a new question arises for network operators: How to juggle with different resource requests, in order to determine the best compromise between **safe** use of the network (e.g. by allocating peak rates) and **efficient** use of the bandwidth (e.g by assuming that users won't start using the bandwidth at once)?

The solution to this problem comes down to non-linear optimization (see [6]). Thus, simulations are performed to test traffic mix and load situations the equipment can cope with and still guarantee the requested QoS (see [9], chapters 3, 5 and 7). But firstly, it is not possible to include all possible traffic classes[1] and secondly, in the real world these traffic classes usually do not behave exactly according to their statistical model. So, implemented CAC's can only be an approximation to the optimal solution of the non-linear problem and parameters are presented to the network operator to adjust the scheme due to her real world requirements.

In [8] one of these parameters is the 'robustness' of the system, which allows the operator to select the multiplexing scheme in a "continuous range from a highly conservative to a very daring " way. However, it is obvious that no optimal solution can be implemented and an operator is needed to adjust the scheme. This adjustment of the parameters will probably not be static. At certain times a very safe resource allocation scheme could be necessary whilst for other times a more generous strategy would be possible.

Here, self-regulating network management should come into play and relieve the operator from this ongoing adjustment of resource allocation parameters.

This cannot be achieved when using the above depicted 'device view' for management purposes. In this case, certain values of management parameters (i.e. link load, discarded cells on a connection) are considered as *critical* and the management station (usually the network operator) receives an alarm when the values are exceeded. This would lead her to correct the selected resource allocation scheme. But which values are critical? How to change the presented parameters? When is it possible to be 'generous' and when is it necessary to be conservative? At least it is probable that by the time we receive the alarms we already have problems and would tend to use a more conservative strategy.

[1] Traffic classes in this sense are characterized by resource requests for a certain amount of bandwidth and QoS.

So, in attempting to relieve the operator of these decisions the key questions to answer are:

- Which parameters to monitor?
- Which values to consider as critical?
- Which actions to perform, after critical values have been exceeded?

It is not possible to answer these questions when using the 'device view'. The equipment itself does not tell us which values are critical with respect to optimal resource allocation.

For this reason we now define a more abstract task of network management, i.e. to support the quality of an ATM connection for the time it exists. Now we ask – and try to answer – which circumstances disturb an ATM connection and what helps to prevent this. Therefore, we are using a 'connection-oriented' view for the definition of management goals.

Before going into a more detailed analysis, certain assumptions regarding the possibilities for a NMS to influence resource allocation have to be discussed.

3 Resource Allocation and Network Management

Although the work of [8] already integrates performance management and admission control, existing equipment does not yet implement corresponding features. Most vendors just began with the release of signaling capable software and are far away from providing features regarding resource allocation adjustment.

Besides the future possibility to influence the admission control, one important point in this work is the renegotiation of the traffic contract for existing connections. Renegotiation is not standardized yet, although it's importance is obvious. It will not be possible to admit a single user to use a large part of the available resources, if the possibility to lessen his resources when contention occurs does not exist.

For public providers, renegotiation will be a sensitive point despite its usefulness. The possibility to get renegotiated must be agreed upon at establishment time and considered with appropriate tariffs. This aspect is considered in this work in the form of 'fairness policies' which have to specify restrictions regarding renegotiation. However, further considerations in this work will show that the potential expense to include renegotiation facilities in public service contracts is justified.

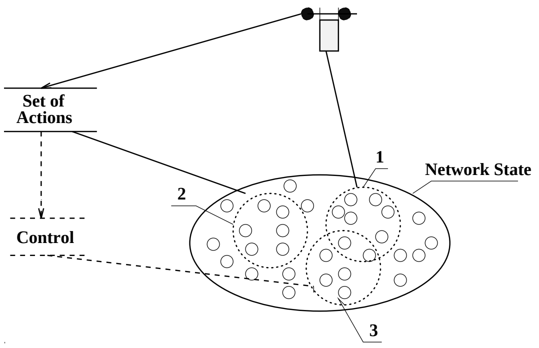

Figure 1: Keeping the network in a 'user-friendly' state.

A further point to consider are available services in the network. Although several services are roughly specified[2], most vendors - especially in the LAN environment - implement their own set of services. These tend to be a 'best effort service' mostly comparable to ABR, and a guaranteed service, which means that a certain peak rate is guaranteed (compare switches from Cisco Systems and Fore Systems). Other vendors (for example GDC[3]) provide multiple VBR Services with different quality assurances and a CBR service. In any case it is possible – and done so in this work for generality – to map the different services on requests for a certain mean rate and peak rate, where only the mean rate is usually guaranteed.

Another important point in this work is the restriction of the resource allocation area to bandwidth allocation questions. The inclusion of requirements regarding cell delay has not been considered so far. The integration should thus be possible, when following the thread of the next section.

4 Constructing self-regulating Network Management

In Figure 1, required components of a self-regulating network management are illustrated. The network state is described by a certain set of management parameters. Part of these, called set 1, presents parameters for the self-regulating mechanism whose task is to keep the network in a state where most[4] of the network users get the requested quality to transport their data. Further on this will be referred to as the 'user-friendly state'. When certain values of these parameters are exceeded, a set of actions has to be initiated to

[2] Variable Bit Rate Service (VBR), Constant Bit Rate Service (CBR). Available Bit Rate Service (ABR) in development. See [2] and [3]

[3] General DataCom

[4] desirable: all

prevent the network from leaving this state. This 'set of actions' needs to modify the configuration of well-known parts of the network, the second set of management parameters. The correctness of the actions taken should be controllable, i.e. a third set of parameters has to be determined whose values clearly prove that we did not perform the wrong actions and the time for the actions was accurate.

We now have to identify the appropriate set of parameters, actions and time to perform these actions, that means 'critical' values of the parameters. In the rest of this paper, we refer to these 'critical values' as 'action values', as they require a certain action.

The parameters are derived from management goals which are obtained when using the 'connection-oriented view', introduced in section 2. First we ask what could happen to an ATM connection and then analyze the possibilities to prevent the identified problem cases. This leads us to an informal description of the required components of the system in Figure 1.

After the theoretical analysis the setting of action values is discussed in section 4.3 and some approaches are presented.

4.1 What could happen to an ATM connection?

With respect to resource allocation problems[5] the following problem areas can be identified for an ATM connection:

1. If the connection is established with signaling, the call setup request could be blocked.
2. The allocated bandwidth for a connection could not be accurate and cells get discarded by traffic policing.
3. After their establishment, connections can lose cells on congested links in the network.

We now analyze each of the three areas to recognize what helps to prevent the occurrence of these troubles.

4.2 Analysis of the Problem Cases

4.2.1 Avoiding Call Blocking

As already discussed, there are many possibilities to minimize call blocking, e.g. optimal VP configuration ([1]), a dynamic CAC by taking previously monitored call intensities and call blocking constraints into consideration ([8]). But even

[5] Problems specific to switch architectures are not considered in this work.

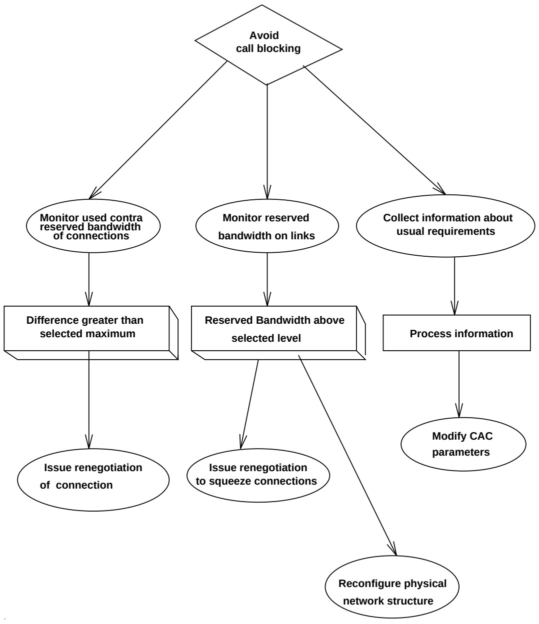

Figure 2: Avoid Call Blocking

when assuming a dynamic admission control, the possibility of call blocking remains. Depending on the area the ATM network is used in, this might not be acceptable. In the LAN area, we usually want everybody to be able to communicate and so call blocking is not acceptable at all. In the WAN area, the primary aim could be to get as much money for the provided bandwidth as possible, and so call blocking could be acceptable depending on the service contract.

Assuming that we want to serve as many customers as well as possible, we have three possibilities to avoid call blocking (compare Fig. 2):

1. Control bandwidth usage

First of all, we can try to avoid the waste of bandwidth by making sure that everybody really uses the allocated bandwidth. We can do this by monitoring the used resources of the connections and compare it with the allocated resources. If the difference is greater than a selected maximum, renegotiation of the connection is initiated.

2. Squeezing connections

When a certain amount of reserved resources on a particular link is reached, indicating that the probability of a call blocking event is very high, renegotiation on selected existing connections is initiated. Thus, we are squeezing existing connections to get room for potential new ones. If no further squeezing of connections is possible and a high call blocking rate is observed, we might have reached a physical limit of our network and a redesign of the network is necessary (similar to splitting of highly loaded Ethernet segments as mentioned previously).

3. Modify CAC

We include the knowledge about *usual* traffic characteristics in our network to modify the CAC respectively.

Discussion

It is necessary to consider some restrictions to these points. It does not make sense to renegotiate every connection. If the usage of the bandwidth is low, we could omit any action on existing connections, which includes the possibility to disturb them. Besides, some kind of services or connections of a special user probably should not get renegotiated at all. Thus, it should be obvious that the set of actions we are looking for has to be restricted by a set of 'fairness policies'.

The actions themselves should be evident. Concerning the control of bandwidth usage, we act as follows: If the used bandwidth of a connection differs more than a value $diff_{max}$ for more than a time $diff_{time}$ from the allocated bandwidth, this connection would get renegotiated to the observed values. If renegotiation is either not possible (paying customer) or not worth (no further usage on the link expected) due to the fairness policies, we omit renegotiation and instead give the customer a report after the connection was released.

As far as squeezing connections is concerned the following considerations apply: The probability of call blocking is very high, when the amount of free resources on a link is less than the *usually* requested mean bandwidth on this link and there are currently less connections active than *usual*. This means that we need an action value for the management parameter $LinkBandwith_{reserved}$ which is dependent on the traffic characteristics in the network. We especially need to know how many connections are usually active on this link and how many resources are usually requested. This issue will be discussed in section 4.3. Again the action to initiate after these values are reached is to renegotiate selected connections based on the policies.

The modification of the CAC is not bound to special management parameter values but is time dependent. By logging the resource requirements appearing and processing them (e.g. time dependent number of requests and amount of resources requested) the parameters of the CAC could be manipulated accordingly (for example: do not accept, respectively lower down, a request for more than 10 % of the physical bandwidth at 1 pm on Mondays as this time we usually have 10 connections each requesting 10 % of the resources; or: at 6 pm on Sundays, it is possible to accept requests for up to 50 % of the resources as this time we usually only have 5 connections each requesting 10 % of the available resources).

The control section depicted in Figure 1 has to validate that no 'call rejection' appeared.

4.2.2 Ensuring Accurate Bandwidth Reservation

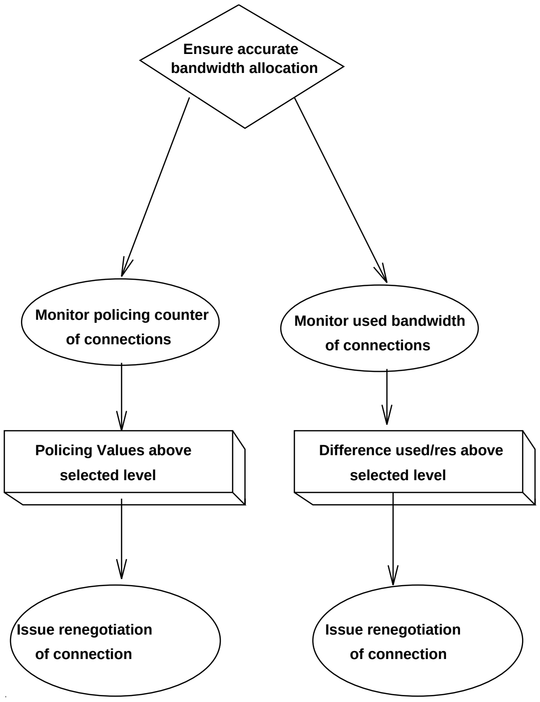

Figure 3: Ensure Accurate Bandwidth Reservation

In section 4.1, only the case of too little allocated bandwidth was considered, as only this case would lead to cell losses for a connection. However, to ensure the correct amount of allocated bandwidth for a connection, we need to consider both cases, i.e. too little and too much allocated bandwidth. So for completeness, the 'too-much' case was added in Figure 3. For the discussion of this branch please refer to section 4.2.

The allocation of too little bandwidth is indicated by a raise of the 'policing counters'. The policing counters available for a connection vary according to the equipment used. For simplicity, they are considered here to be counter for 'discarded cells'.

One has to find a value for the amount of discarded cells, that indicates problems on this connection. It would be reasonable to set the action value in relation to the amount of cells transported with this connection. As this is not possible, we again have to try to find a good approach in using a relationship to *usually* transported cells on a connection of a particular link. The action to perform when this value is exceeded is to inform the user and to allocate more bandwidth to this connection, i.e. again perform a renegotiation.

The control part of the searched, self-regulating system, as depicted in Figure 1, is used to adjust the selected relationship. This means that it has to validate that the discarding of cells is transparent to the user until the system 'acts', in particular, the control has to be performed manually after a user told that she has some problems.

4.2.3 Avoiding Congestion

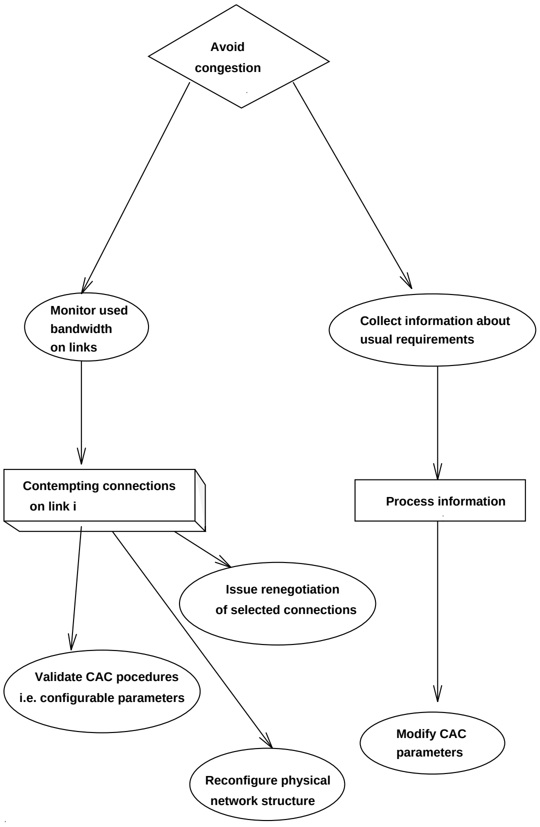

Figure 4: Avoiding Congestion

The problem of congestion always exists when a progressive bandwidth sharing strategy is used. The effects and handling of congestion in switches differ depending on the equipment used (compare e.g. [7]). We assume that congestion is problematic for the traffic in the network in any case and should therefore be avoided where possible.

Congestion is indicated by some increase in the used bandwidth on the output links and depending on the used equipment on certain buffer values. For generality, we only consider the amount of used bandwidth so far. We have to determine a limit, which surely points on an approaching congestion. This is quite complex, as the increase in the use of bandwidth could be very sudden and would make it impossible to act in time.

As previously, a possible action value is related to usually active connections and their requested resources. If there are currently less connections active than usual and more connections are configured than active (i.e. we have idle connections) the probability of congestion is very high if the resources left on a link are less than the mean resources which the idle connections request. However, this is only one possible indication of congestion. Further approaches, which include vendor specific management parameters as well, are topic for further research.

Again, the action to perform is the renegotiation of existing connections according to the already introduced fairness policies. If no renegotiation is possible and congestion appears often the physical redesign of the network structure should be considered.

After such an occurrence, we should remember this situation and use the knowledge to manipulate the admission control accordingly (second branch of Figure 4). That means a report has to be generated, where the time and traffic configuration which led to the (potential) congestion is registered. The CAC then has to be adapted in order not to accept this traffic mix (possibly in conjunction with the sources who send the traffic) at this time again.

The control part in this case will be a negative message, i.e. there was a congestion on a particular link in spite of the renegotiation of connections. Note that the detection of a potential congestion can be seen as the control part for the second branch in Figure 4.

4.2.4 Summary

Due to the analysis above the self-regulating system can be complemented as shown in Figure 5.

The network state is continuously written down in a history database (A). This history information is processed (B) and leads to the definition of action values (C) for the set 1 of parameters[6]. When these thresholds are reached, a well-defined set of action is performed (D), manipulating the second set of parameters[7]. Afterwards, the correctness of the operations is checked

[6] An example element of set 1 is the parameter $LinkBandwidth_{reserved}$

[7] E.g. manipulate allocated resources of existing connections. The configuration parameter 'allocated resources' is thereby an example set-2-parameter

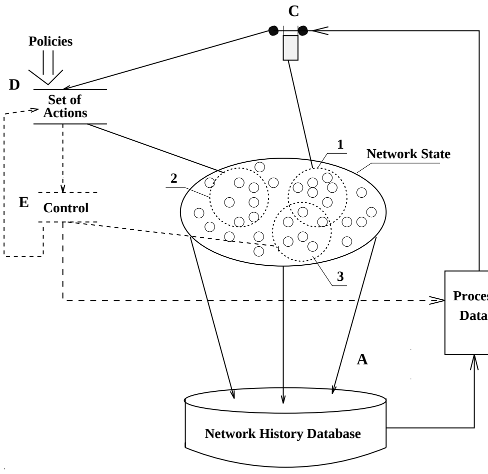

Figure 5: Basic elements of the self-regulating system

by the control part (E), in validating values of the third parameter set[8]. A negative response[9] of the control part indicates necessary modifications of the parts B and D.

The processing of history information is discussed in the next section. The necessary set of actions was informally described in above considerations. The key actions were identified to be the renegotiation of traffic contracts and the modification of CAC parameters. Neither is possible with existing equipment, renegotiation is not even standardized. A more detailed specification of the action set has to be performed as soon as these features are available.

However, quick renegotiation in upcoming problem situations, as described above, requires that candidates are marked already at call setup time. I.e. if we detect the necessity to renegotiate in expectation of a congestion, we must already know which concrete connections are candidates subject to the given fairness policy.

The control part (E) in Figure 5 is either implicit given (a user telling us, that problems occur) or can be explicitly implemented (e.g. in terms of the number of call rejections). The automatic manipulation of the parts B and D in Figure 5 after negative responses is therefore case dependent.

The most important point and the base for further work is the collection of relevant traffic data and the processing of this data to get action values. This will be depicted in the next section.

[8] E.g. number of call rejections

[9] E.g. call rejection appeared

4.3 Setting Action Values

The above considerations are based on the assumption that *usual* values really exist, i.e. that a certain regularity is detectable in the network traffic.

Action values are thus based on the following:

1. Collect information about the traffic in the network and map it to usual values, i.e. define the granularity for this mapping.

2. Relate the usual values with the needed action values.

The following usual[10] values must be estimated:

- $active_{av}$
  – Average number of usually active[11] connections on this link.

- $conf_{av}$
  – Average number of usually configured[12] connections on this link.

- $mean_{av}$
  – Average amount of mean bandwidth usually requested by connections on this link.

- $cells_{av}$
  – Average number of cells transported with one connection on this link.

- dur_{av}
  – Average duration of a connection on this link.

We need to select action values for the following set of parameters:

- $LinkBandwidth_{used}$
  – Indication of congestion

- $LinkBandwidth_{reserved}$
  – Indication of call blocking

- $diff_{max}, diff_{time}$
  – The maximum allowed difference between used resources and allocated resources and the time this difference is tolerated.

- $discard_{max}$
  – The maximum amount of discarded cells

[10] 'Usual' will here be estimated with average values. It is of course possible to work with maximum values as well.

[11] Active thereby means: 'Currently transporting cells'.

[12] The distinction between active and configured connections is necessary, as 'configured' does not necessarily imply 'active'.

on a connection before the user recognizes problems.

Due to the considerations in section 4.2 the relationships are set as follows.

4.3.1 Call blocking indication

$$\begin{aligned} LinkBandwidth_{reserved} &\geq pb - av_{mean}, \\ \text{subject to} & \\ curr_{conf} &< av_{conf} \end{aligned}$$

where $curr_{conf}$ equals the number of currently configured connections on a link and pb equals the physical bandwidth of this link.

4.3.2 Not-accurate-bandwidth indications

The first case to consider here is the 'too-much' case. We need to decide, when it is worth to renegotiate a connection in order to save bandwidth. This is at least the case, when corresponding bandwidth savings on all existing connections would free resources for a new connection. Therefore we relate the required action value with average active connections and usually requested resources by a connection in this link:

$diff_{max} = \frac{av_{mean}}{av_{active}}$

To determine the time a connection is allowed to exceed the maximum difference between allocated and used bandwidth, we must consider tolerated idle times and fluctuations. Thus it is reasonable to define the value $diff_{time}$ in terms of the usual duration of connections on this link:

$diff_{time} = \frac{av_{dur}}{c_1}$

where c_1 is a constant depending on personal requirements regarding bandwidth savings. c_1 has to be greater, if call blocking occurs often and vice-versa. A value of '2' seems a good approximation as starting point.

The 'too-little' case on contrary should set the maximum amount of discarded cells on a connection in relation to usually transported cells on connections of this link.

$discard_{max} = \frac{av_{cells}}{c_2}$

Again an initial value would be chosen for c_2 (e.g. 10) and has to be adapted by the control part of our model, i.e. the user calling us because he has problems.

4.3.3 Congestion indication

$$\begin{aligned} LinkBandwidth_{used} &\geq pb - av_{mean}, \\ \text{subject to} & \\ curr_{active} &< av_{active} \\ & \wedge \\ curr_{active} &< curr_{conf} \end{aligned}$$

where $curr_{active}$ equals the number of currently active connections on this link and $curr_{conf}$ equals the number of currently configured connections on this link. This approach is based on the following assumptions:

1. The number of configured connections is not necessarily equal to the number of active connections.
2. If connections become active they transport their cells with at least the agreed mean rate.
3. The considerations explained in section 4.2.

The suggested relationships should give indications on choosing reasonable action values. It is necessary to adapt them in concrete production environments. However, this work stresses the usefulness of these values and thereby the urgent need to approximate them.

The following section discusses preliminary implementation experiences.

5 Implementation Experiences

Figure 6 shows the test network at the Leibniz Computing Center in Munich and the connection to the HD012 project[13]. We used the network to drive performance tests in various configurations with switches from Netcomm[14] and Fore Systems. The performance, especially via the 34 Mbps link, was acceptable, when the end systems were able to shape the submitted traffic to a specified peak rate. This was not the case at the beginning of the tests and thus there were heavy throughput problems due to discarded cells.

The workstation denoted 'OpenView' in Figure 6 is used to implement the self-regulating system proposed in this work. The collection of traffic information is implemented with the SNMP-based platform HP OpenView. Several critical points arise here. Vendor specific MIBs have to be used, as standardized MIBs [10] are not yet available and it is not clear if they will support the detailed information requested in this work. To cope with the multi-vendor environment we implement a generic MIB and take necessary vendor specific details into consideration.

[13] The HD012 was a project to evaluate technologies for wide area networks. The project was conducted by the Regional Computing Center (RRZE) of the University Erlangen-Nuernberg and funded by the DFN (German Research Network) with means of the German Government.

[14] Now General DataCom

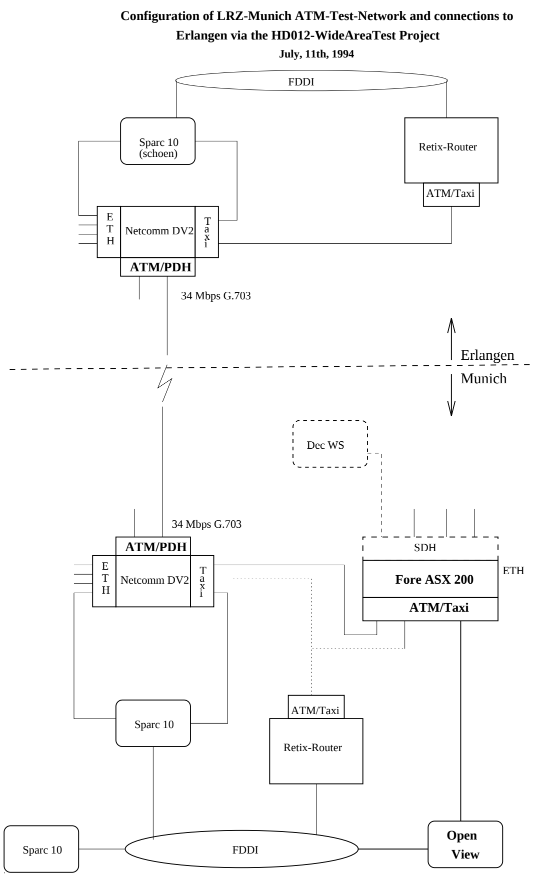

Figure 6: Test Environment

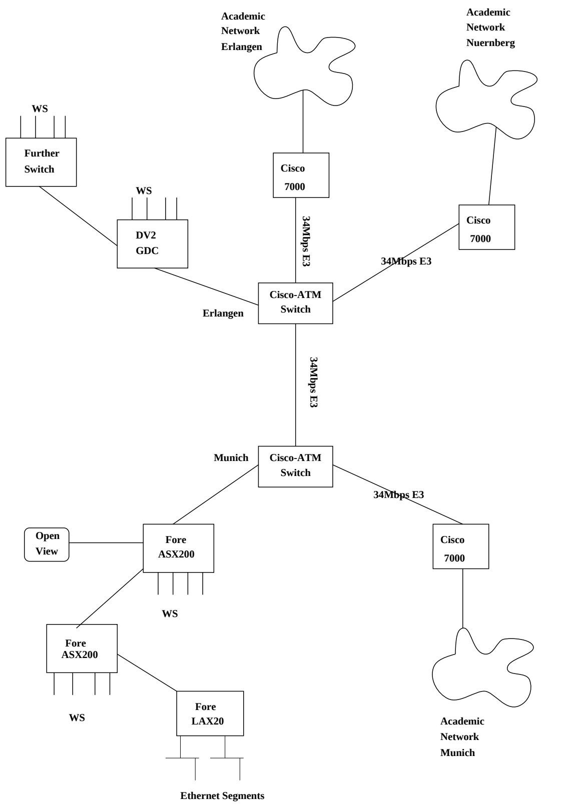

Figure 7: RTB production environment

The information is collected using an event-driven technique. This means, whenever one of the interesting parameters changes a value, an event is generated to log the corresponding information. The delay detected when querying one switch, varies round one second and will certainly increase when the network gains complexity in components and traffic.

Figure 7 presents the planned configuration for the RTB project[15]. The aim of the RTB project is to evaluate applications with high bandwidth demands in the wide area. ATM Switches and Routers from Cisco Systems are used to provide the infrastructure as depicted in Figure 7. The applications provide a suitable traffic mix to gain experience in the the detection of regularity and the usage of action values.

The resulting delay for the collection of information will be evaluated. It is easily detectable that the appropriate environment to implement the suggested mechanism would be CMIP, where logging and threshold checking is done by the agents. The implementation in an SNMP environment can only reach the goal to a smaller extent. But solutions for SNMP have to be found, as this is the management protocol supported by most vendors. These solutions will be considered in further research.

[15] Regional Testbed project. Funded by the DFN association with means of the German government. Compare presentation of Peter Kaufmann, track N6, "The Implementation of a High Speed Network for the DFN Community".

6 Conclusions and Outlook

To be able to provide efficient management for a demand-oriented ATM network and prevent simultaneously the occurrence of unsatisfying network states, a goal is certainly to achieve a self-regulating network management. This work suggested to collect knowledge about usual traffic characteristics, and use this knowledge as a base for an appropriate configuration of resource allocation parameters.

We presented a model, including the necessary parameters to (i) monitor the behaviour of the network, (ii) to act on, and (iii) to validate whether the performed actions have achieved the desired goal. The set of actions to be performed is also dependent on some fairness policies, which are user specific. Control mechanisms were introduced to validate whether the performed actions had achieved the expected result. Although all elements of the model were discussed, we concentrated on the specification of parameters to monitor the behaviour of the network and the definition of action values.

Our further work will focus on (i) testing the feasibility of the specified parameters in production environments, especially the regional testbed, (ii) the refinement of parameters (i.e., using other than average values) if appropriate, (iii) a formal description of the actions to be performed under certain fairness constraints (i.e., policies). Besides, a detailed development of the control part and its evaluation is necessary.

Acknowledgments

The authors wish to thank the members of the Munich Network Management (MNM) Team for helpful discussions and valuable comments on previous versions of the paper. The MNM Team is a group of researchers at the Leibniz Computing Center Munich and the Munich Universities. It is directed by Prof. Dr. Heinz-Gerd Hegering. We gratefully acknowledge in particular Victor Apostolescu, Thomas Kaiser, Ulrich Katzenschwanz and Bernhard Neumair who provided valuable suggestions and advice.

References

[1] Nikos G. Aneroussis, Aurel L. Lazar, " Managing Virtual Paths on Xunett III: Architecture, Experimental Platform an Performance ", Proc. 5th IFIP Integrated Network Mgmt. Symp. May 1995.

[2] ATM Forum UNI 3.0 Specification, September 1993.

[3] ATM Forum Traffic Management Specification Version 4.0, Draft Version 2.0, February 1995.

[4] Dominique Gaiti, "Introducing intelligence in distributed systems management", Computer Communications, vol. 17, no. 10, pp. 729-738, October 1994.

[5] Jay M. Hyman, Aurel L. Lazar, Giovanni Pacifici, "Real-Time Scheduling with Quality of Service Constraints", IEEE Journal on Selected Areas in Communication, vol. 9, pp. 1052-1063, September 1991.

[6] Jay M. Hyman, Aurel L. Lazar, Giovanni Pacifici, " A Separation Principle between Scheduling and Admission Control for Broadband Switching", IEEE Journal on Selected Areas in Communication, vol 11, pp. 605-615, May 1993.

[7] Robert Mndeville, "The ATM Stress Test", Data Communications International, March 1995, pp 68-82.

[8] Giovanni Pacifici, Rolf Stadler, " Integrating Resource Control and Performance Management in Multimedia Networks ", CTR Technical Report 388-94-35, Center for Telecommunications Research, Columbia University, August 1994.

[9] Martin De Prycker, "Asynchronous Transfer Mode", Ellis Horwood, 1993.

[10] M. Ahmed, K. Tesink, " Definition of Managed Objects for ATM Management, Version 8 ", Internet Draft RFC1695, August 1994.

[11] Alex Allister Shvartsman, "Dealing with History and Time in a Distributed Enterprise Manager", IEEE Network, vol.6, pp. 32-41, November 1993.

Author Information

Anja Schuhknecht is a research analyst at the Leibniz Computing Center in Munich. She studied computer science at the Technical University of Munich and obtained her diploma in 1992. Since then she is working at the Leibniz Computing Center with main responsibilities in high speed networking and multi media applications. Her special interest is ATM network management.

Gabi Dreo received B.S. and M.S. degrees in computer science from the University of Maribor, Slovenia. Afterwards, she was a research staff member at the department of Computer Science of the University of Munich. Soon she will receive her Ph.D from the University of Munich. In January 1995 she joined the Leibniz Computing Center where she is engaged in several network management projects.