EUROCONTROL

ATNP/WG2/
WP/197
29 January 1996

AERONAUTICAL TELECOMMUNICATIONS NETWORK PANEL

WORKING GROUP TWO

Brisbane 5.2.96-9.2.96

Results of Congestion Management Simulations

Presented By Henk Hof

Prepared by Leen Goossens

SUMMARY

Eurocontrol has undertaken simulation exercises in order to prove the need for congestion management in the ATN and to examine the advantages and drawbacks of several congestion management strategies. This paper presents the results of these simulation exercises. As a result of this work, it is possible to justify fully the need for and advantages of received based congestion avoidance. There is a considerable improvement in overall transit delay as a result of implementing the proposed algorithm, and this working paper recommends the inclusion of the proposed congestion avoidance algorithm in the CNS/ATM-1 Package SARPs.

DOCUMENT CONTROL LOG

SECTION	DATE	REV. NO.	REASON FOR CHANGE OR REFERENCE TO CHANGE
	29-Jan-96	Issue 1.0	

TABLE OF CONTENTS

1. Introduction

1.1 Background and Objectives

Eurocontrol has undertaken a series of simulation exercises in order to prove the need for congestion management and to examine the advantages and drawbacks of several alternatives. This work was undertaken as an action from WG2, following earlier Eurocontrol papers that argued that Congestion Management procedures were an essential part of the ATN Specification.

The overall objectives of this work have been to demonstrate the need for Congestion Management procedures in the CNS/ATM-1 Package SARPs, and to select the most appropriate Congestion Management Procedure. In order to achieve this, a set of simulation exercises has been specified and performed. The objectives of the simulation exercises are to:

1. investigate each of the alternatives, including non-use of Congestion Management
2. report on their advantages and drawbacks,
3. recommend the best strategy for the CNS/ATM-1.

1.2 Scope

The following congestion management strategies are defined in section three of this paper:

- Receiving Transport Layer Congestion Avoidance,
- Receiving Transport Layer Withdrawing Credit,
- Sending Transport Layer Backoff.

Simulation Exercises have been performed to investigate each of the above Congestion Management strategies, over a range of different parameter values. The specifications of these exercises are included as an annex to this paper, and analysis of the results included in section 5 of this paper. The validation tools itself are presented in section 4.

This paper presents the results of these exercises.

1.3 References

[CNS/ATM-1 draft Internet SARPs]	CNS/ATM-1 draft Internet SARPs, Version 3.0
[DED1/EAS3/STA/DOC/95_05]	Conventions in support of validation exercises specifications
[DED1/EAS3/STU/DOC/21]	European ATN Simulation Strategy
[DED1/EAS3/STU/REQ/01]	Transport Layer Simulation Models Requirements Specification
[DED1/EAS3/STU/REQ/02]	CLNP and 8208 Simulation Models Requirements Specification
[DED1/EAS3/STU/DCI/01]	TP4 Model User Manual
[DED1/EAS3/STU/DCI/02]	CLTS and COTS Architectural Design

[DED1/EAS3/STU/DCI/08] CLNP Model Architectural Design

[DED1/EAS3/STU/DCI/10] CLNP Model User Manual

2. Executive summary

Connectionless Internets, such as the ATN, are vulnerable to poor performance during periods of congestion. In fixed networks, this can be avoided by simply ensuring that more network capacity is available than is ever needed. However, the ATN is constrained by the limited bandwidth available to air/ground data links. Congestion on such networks cannot be simply avoided by over-provision of networking resources, and, in order to ensure that the ATN will meet the Quality of Service requirements of safety related applications, it is believed essential to implement, on an ATN wide basis, effective Congestion Management procedures.

Eurocontrol has investigated the impact of congestion through simulation exercises. The effect of three candidate Congestion Management algorithms has been simulated, along with the use of no Congestion Management algorithm, in order to provide a baseline against which the results can be interpreted. These candidate algorithms are specified in section 3, and the results are analysed in section 5 of this paper. From analysis of the results, it can be concluded that there is very significant value in mandating the use of a Congestion Management algorithm in CNS/ATM-1 Package, and that such algorithms should be viewed as essential features of the ATN Internet.

The simulation results show that there is the potential for an order of magnitude improvement in transit delay during periods of network congestion, when the best Congestion Management algorithm is chosen, compared with the situation where no Congestion Management algorithm is implemented.

From this work, it is possible to conclude that the algorithm specified in section 3.1, and known as the “Receiving Transport Layer Congestion Avoidance” algorithm, is the best algorithm out of those investigated. This algorithm is implemented by the receiving transport layer entity and depends upon information supplied through the CLNP Congestion Experienced (CE) bit.

The results of the simulation exercises show that this algorithm has significant advantages, provided that the threshold for setting the congestion experienced flag is very small. The benefits of implementing this algorithm may be summarised as:

- there is a higher throughput,
- the end-to-end delay is significantly lower,
- the buffer load is significantly lower,
- fewer packets are discarded,
- fewer packets need to be retransmitted.

It is therefore **recommended** that WG2 adopts the Change Proposal contained in WP231, and consequently includes the specification of this algorithm in the ATN Internet SARPs.

It should be noted that earlier concerns over the impact of mobility have not been overlooked in this work. WP159 and WP160 discusses this topic at length. However, these concerns apply to Congestion Management by the sending transport layer entity, and as the unambiguously best algorithm was one that is implemented by the receiving transport layer entity, it was not believed necessary to investigate this aspect further. Indeed another important advantage of the proposed algorithm is that it is not expected to be affected by mobility considerations.

However, simulation work is still continuing in respect of determining the best algorithm for calculating the TP4 retransmission timer through measurement of the round trip delay. This is believed necessary for any Wide Area Network and essential for the ATN, where wide variations in transit delay between the various air-ground data links are a known characteristic. This algorithm is expected to be impacted by mobility, as discussed in WP160. However, this can now be regarded as a separate issue from Congestion Management.

Work is also continuing in simulations and, in particular, to assess whether there is a need to modify the algorithm for setting the congestion experienced bit, in order to take into account the priority of the DT PDU.

It is hoped to present conclusions from the investigation of both of these topics at the next Working Group Meeting.

3. Congestion Management Strategies

As part of the work of investigating Congestion Management, a number of alternative Congestion Management strategies have been defined and investigated. The following strategies are defined in this section:

- Receiving Transport Layer Congestion Avoidance;
- Receiving Transport Layer Withdrawing Credit; and,
- Sending Transport Layer Backoff

3.1 Receiving Transport Layer Congestion Avoidance

In this strategy, Congestion Management is exercised by the receiving transport entity. The receiving transport entity tries to avoid congestion by decreasing the size of the window that it advertises to the sending transport entity from the moment the network load increases slightly and by increasing this size when the network load decreases[1]. The receiving transport entity tries to find a value for the size of the advertised window so that optimal use will be made of the underlying network without that the network gets heavily loaded. The buffer load in each IS must be kept sufficiently low so that end-to-end delay will not suffer but at the same time buffers may not empty otherwise no optimal use is made of the network.

At the network layer, ISs set the *congestion experienced* flag in packets traversing them when their buffers are loaded. When the destination network entity receives a packet with the *congestion experienced* flag set, it must inform the transport entity. In this way the receiving transport entity can monitor the load of the network. Network layer and transport layer must co-operate to avoid congestion.

3.1.1 Setting of the congestion experienced flag

If an NPDU arrives at an IS, the IS examines the depth of the output queue selected for that NPDU. If the depth of the selected output queue exceeds a known threshold, α, the IS sets the *congestion experienced* flag before forwarding the NDPU to the destination. The *congestion experienced* flag is a flag in the QOS Maintenance Parameter option header.

When the *congestion experienced* flag is conveyed to a destination network entity, this entity conveys the *congestion experienced* information to the destination transport entity by local means. This can be done by using the N-REPORT primitive.

---

[1] The advertised window is the window that the receiving transport entity advertises to the sending transport entity to indicate the packets that it is willing to accept from the sending transport entity.

3.1.2 Adjustment of the advertised window

- Initialisation of the advertised window

  The initial value of the dynamic value of the window that is advertised to the sending transport entity (W_0) should have a locally configurable upper bound. This window is sent to the sending transport entity in the next credit (CDT) field transmitted.

- Sampling Period

  All receiving transport entities maintain a fixed value for the size of the advertised window until the next 2*size(advertised window) DT TPDUs arrive since the last CDT field was transmitted by the receiving transport entity.

- Counting of Received TPDUs in a Sampling Period

  All receiving transport entities maintain a count, N, equal to the number of TPDUs received, and a count, NC, equal to the total number of TPDUs received which had the CE flag set in the associated N-REPORT primitive. All types of TPDUs are included in the counts for N and NC, not just DT TPDUs.

- Action upon the end of a Sampling Period

  All receiving transport entities take the following action at the end of each sampling period:

  1. If the count NC is less than 50% of the count N, the receiving transport entity increases the size of the advertised window by adding one up to a maximum based on the local buffer management policy. Otherwise, it should decrease the size of the advertised window by multiplying by β (β should be a value smaller than one). The size of the advertised window can go to a minimum of 1.

  2. Reset N and NC.

  3. Transmit the new size of the advertised window in the next CDT field sent to the sending transport entity.

3.2 Receiving Transport Layer Withdrawing Credit

As with the previous congestion management technique, in this strategy, congestion management is exercised by the receiver of the data who is informed of congestion in the network by the network entity. A sliding window flow control strategy is used for the window that is going to be advertised to the sender of the data. At the network entity ISs inform each other and ESs that congestion is experienced through the CLNP Congestion Experienced (CE) bit.

Initially the size of the advertised window is based on the local buffer management policy. When the receiving transport layer is informed that congestion is experienced at the network layer, it gradually closes the advertised window. This is done by adjusting the lower window edge of the advertised window as packets are received, but the upper window edge of the advertised window cannot be changed until all packets that are in the current advertised window are accepted. After all packets are accepted that were in the advertised window, those packets are acknowledged and a window size of one is advertised. Afterwards the window size may be increased by one each time N packets successfully arrive where N is a multiple (α) of the size of the advertised window and, the definition of “successfully arrive” includes a requirement that the *congestion experienced* flag is not set for those packets. The advertised window will increase until a maximum is reached. This maximum depends on the local buffer management policy.

3.2.1 Setting of the congestion experienced flag

This is the same as 3.1.1 Setting of the congestion experienced flag.

3.2.2 Adjustment of the advertised window

When a transport entity exercises the *Receiving Transport Layer Withdrawing Credit* congestion management technique, transport connection can be into one of three phases:

- a first state (NCE = No Congestion Experienced) where no congestion is experienced: the size of the advertised window is based on the local buffer management policy. A transport connection is in this state at the beginning of the transport connection and following the third state.

- a second state (CE-CAW: Congestion Experienced - Closing Advertised Window) where the receiving transport layer gradually closes the advertised window because it is informed that congestion is experienced at the network layer. This is done by adjusting the lower window edge of the advertised window as packets are received, but the upper window edge of the advertised window can't be changed until all packets that are in the current advertised window are accepted.

- a third state (CE-OAW: Congestion Experienced - Opening Advertised Window) is entered after the second state, when all packets that were in the current advertised window were accepted. The size of the advertised window becomes one and the size of the advertised window is increased by one each time N packets successfully arrive where N is a multiple (α) of the size of the advertised window and, the definition of *successfully arrive* includes the requirement that the *congestion experienced* flag is not set for those packets.

Schematically this becomes:

```
state = NCE;
                                          /* curr_adv_window is the window that is currently advertised */
                                        /* new_adv_window is the window that is going to be advertised */
low_edge(curr_adv_window) = 0;
size(curr_adv_window) = size(new_adv_window) = saw;
                                              /* saw is based on the local buffer management policy */
up_edge(curr_adv_window) = saw;
while ( transport connection exists )
{
        if ( packet arrives )
        {
                if ( ce flag is set for this packet )
                {
                        state = CE-CAW;
                }
                else if ( state = CE-OAW )
                {
                        n_pack_succ_arrived = n_pack_succ_arrived + 1;
                        if ( n_pack_succ_arrived == α * size(new_adv_window) )
                        {
                                size(new_adv_window) = size(new_adv_window) + 1;
                                n_pack_succ_arrived = 0;
                                if ( size( new_adv_window = saw ) )
                                        state = NCE;
                        }
                        else
                        {
                                size(new_adv_window) is not adjusted;
                        }
                }
        }
        if ( AK TPDU needs to be sent )
        {
                low_edge(curr_adv_window) = YR-TU-NR;
                          /* YR-TU-NR = sequence number indicating the next expected DT TPDU number. */
                switch (state) {
                case NCE:
                        size(curr_adv_window) = saw;
```

```
                        up_edge(curr_adv_window) =
                                    (low_edge(curr_adv_window) + size(curr_adv_window))%NF;
                                                          /* NF equals 2^7 when normal formats
                                    are used for the sequence numbers, otherwise it equals 2^31 */
                        break;
                case CE-CAW:
                        size(curr_adv_window) =
                                (up_edge(curr_adv_window)-low_edge(curr_adv_window)+NF)%NF;
                        if ( size(curr_adv_window) == 0 )
                        {
                                size(curr_adv_window) = 1;
                                up_edge(curr_adv_window) = (low_edge(curr_adv_window) + 1)%NF;
                                if ( size( curr_adv_window == saw ) )
                                        state = NCE;
                                else
                                {
                                        state = CE-OAW;
                                        n_pack_succ_arrived = 0;
                                        size(new_adv_window) = size(curr_adv_window);
                                }
                        }
                        else
                        {
                                up_edge(curr_adv_window) is not adjusted;
                        }
                        break;
                case CE-OAW:
                        size(curr_adv_window) = size(new_adv_window) ;
                        up_edge(new_adv_window) =
                                    (low_edge(new_adv_window) + size(new_adv_window))%NF;
                        break;
                }
        }
}
```

3.3 Sending Transport Layer Backoff

The third strategy specified here, specifies a congestion management technique that can be applied by the sending Transport Entity. This algorithm is applied when a packet needs to be retransmitted because the retransmission timer expired or when the network layer signalled congestion (upon receiving an Error NPDU indicating congestion). Congestion management is then exercised by the sender of the data. A sliding window flow control strategy is used.

The sliding window flow control strategy is based on the transport layer keeping track of two different flow control windows:

- the *advertised* window: this is the window advertised by the receiver of the data. It indicates the amount of data that the receiver is willing to accept.

- the *congestion* window: this window is chosen by the sender of the data. It indicates the amount of data the sender will send before an acknowledgement is received. This window should always be a subwindow of the advertised window and the lower window edge of the congestion window should always equal the lower window edge of the advertised window.

Congestion management is exercised by controlling the size of the congestion window. When a packet needs to be retransmitted, it is assumed that the packet got discarded due to congestion. The size of the congestion window is then reduced, i.e. the credit of the congestion window becomes 1. The credit will again be increased when packets have been successfully transmitted, i.e. after acknowledgements have been received.

This congestion management strategy is based on the assumption, that packet loss is mostly due to congestion and not due to the packet having been damaged. A second assumption is that a packet is lost when no acknowledgement is received before the retransmission timer expires. This is not always the case. If the retransmission time is too small, the timer will expire before the acknowledgement is received. Therefore it is very important that a good value is chosen for the retransmission time. This value may not be too small, but it may not be too large either, otherwise performance will suffer under it.

This congestion management technique is also activated when the network layer signalled congestion. When an IS needs to discard packets due to congestion, it will send an Error NPDU indicating congestion to the sender of the data, for each discarded packet that is a Data NPDU and that has the ER flag set to allow Error Reports. As an option, the network entity can also send an Error NPDU indicating congestion when it experiences congestion (before it actually discards data). When the network entity at the sending side receives the Error NPDU, it will inform the sending transport entity.

3.3.1 Sending of an Error NPDU indicating congestion

If a Data NPDU arrives at an IS, the IS examines the depth of the output queue selected for that NPDU. If the depth of the selected output queue exceeds a known threshold, α, the IS will send an error NPDU indicating congestion to the sender of the Data NPDU. An IS will also send an Error NPDU indicating congestion for each packet that is discarded due to congestion that is a Data NPDU and that has the ER flag set to allow Error Reports[2].

The value α can be configurable as a parameter for different subnetwork types.

When the Error NPDU is returned to the originating network entity, this entity should convey the Error information to the originating transport entity by local means, for example by using the N-REPORT primitive. The Error information that is passed on to the transport entity should include the source and destination NSAPs of the NPDU that caused the Error NPDU to be generated. These addresses can be extracted from the Data NPDU header contained in the Error NPDU. In this way subsequent congestion management actions can be restricted to the impacted transport connections.

3.3.2 Adjustment of the congestion window

The sliding window flow control strategy works as follows:

The lifetime of a transport connection is divided into phases.

A phase starts:

- when the connection is established,
- when a packet needs to be retransmitted because the retransmission timer of the packet expired (it is assumed then that the packet is lost),
- when the network layer signalled congestion (see section 3.3.1 Sending of an Error NPDU indicating congestion).

In the beginning of a phase the size of the congestion window equals one packet. Each time an acknowledgement arrives, the size of the congestion window is increased by the number of packets that are acknowledged. In this way the size of the congestion window exponentially increases until an upper bound is reached. The upper bound consist of the minimum of the size of the advertised window and a certain threshold. The threshold equals half of the size of the congestion window at the end of the previous phase. The size of the congestion window can grow past the threshold but much

---

[2] Returning an Error NPDU for every discarded packet is not efficient, because that may result in further congestion. However ISO/IEC 8473 and the ATN Manual mandate that error reports are sent in certain circumstances, so that the number of Error Reports sent cannot be limited because congestion is experienced.

more slowly. The size of the congestion window is then increased by one each time “size(congestion window)” acknowledgements are received.

Schematically this becomes:

```
size(cong_win) = 2 * α;
while ( transport connection exists )
{
    /* Beginning of a phase */
    threshold = max( size(cong_win ) / 2, 1 );
    size(cong_win ) = 1;
    ackrcvd = 0;
    while ( no retransmission timer expires )
    {
        if ( (acknowl. arrives ) && ( size(cong_win) < size(advertised_win) ) )
        then if ( size(cong_win) < threshold )
            then size(cong_win) = min(  size(cong_win) + #packets_ackn,
                                        threshold,
                                        size(advertised_win) );
            else
            {
                ackrcvd = ackrcvd + #packet_ackn;
                if ( ackrcvd > size(cong_win) )
                then
                {
                    ackrcvd = ackrcvd - size(cong_win);
                    size(cong_win) = min(size(cong_win)+1,
                                         size(advertised_win)  );
                }
            }
    }
    /* End of a phase */
}
```

α is the initial value for the threshold.

4. Simulation Validation Tools

4.1 Simulation tool

Within EUROCONTROL's ATNIP Project, OPNET has been selected as the main simulation tool. This choice has also been made by other ATN simulation participants. OPNET is a sophisticated workstation-based environment for the modelling and simulation of communication systems, protocols, and networks.

4.2 Simulation models

4.2.1 CLNP Model

The Connectionless Network Protocol (CLNP) Model is based on the concepts developed in ISO 8473: 1992, ISO 8348: 1993, ATN Manual. It models the transfer of connectionless Network Service Data Units (NSDUs) from a source Network Service Access Point (NSAP) to a destination NSAP without maintaining any logical connection or relationship between these source and destination NSAPs.

The CLNP Model contains a static FIB. This is just a data store containing forwarding instructions giving for each known NET, the NET of the next hop.

4.2.2 TP4 Model

The transport protocol class 4 (TP4) Model operates over the connectionless network protocol. This model, based on the concepts developed in ISO 8073:1991, ISO 8072, ATN Manual, is viewed as a set of procedures defined in terms of:

a) the interactions between peer transport entities through the exchange of transport protocol data units (TPDUs);

b) the interactions between a transport entity and the transport service user in the same system through the exchange of remote and stream interrupts accompanied by Information Control Interface structures;

c) the interactions between a transport entity and the network service through the exchange of stream interrupts accompanied by an ICI structure.

The primary aim of the TP4 model is to provide an OPNET implementation of rules of communication expressed in terms of the procedures to be carried out by peer entities at all time of communication. This model implements the principal features of TP4 protocol and is intended to provide a basis for use in testing and sizing ATN systems and networks.

The TP4 model permits a duplex exchange of information between the pair of TS users supporting the connection. The TP4 model supports the simultaneous existence of any number of transport connections within a single host model. Each connection is identified by a source and destination references and an initiator host network address.

5. Simulation Results

This section presents the results of the simulation exercises specified in annex A.

Table 5-1 summarises the results in case optimal values for the parameters of the different congestion management techniques were used; the results were obtained for each congestion management technique for an equal level of congestion.

It is clear that performance is significantly improved by using one of the receiving transport layer congestion management algorithms. The end-to-end delay and the buffer load significantly decrease while the throughput increases. There are almost no packets discarded due to congestion and few packets need to be retransmitted.

	mean end-to-end delay (seconds)	mean throughput (i.e. the number of successfully transferred packets per sedond without taking into account retransmissions)	power (i.e. throughput / end-to-end delay)	percentage of packets that are discarded due to congestion	ratio of number of successfully transferred packets to the number of transmitted packets without taking into account retransmissions	ratio of number of successfully transferred packets to the number of transmitted packets taking into account retransmissions	mean buffer load taken at a congested IS	ratio of the number of Error NPDUs sent that indicated congestion to the number of NPDUs sent (Data and Error)
No congestion management strategy	3.2162	4.4888	1.9357	0.0700	0.9766	0.8883	166863	
Receiving Transport Layer Congestion Avoidance	1.1406	4.6848	4.1073	0.0003	0.9914	0.9904	71872	
Receiving Transport Layer Withdrawing Credit	0.9799	4.6983	4.7947	0	0.9888	0.9888	61427	
Sending Transport Layer Backoff	3.9124	4.702	1.2018	0.0371	0.9879	0.9394	187201	
Sending of an Error NPDU indicating congestion	2.4627	4.6881	1.9036	0	0.9898	0.9898	121054	0.2352

Table 5-1 Simulation Results with Optimal Parameter Settings

In Table 5-1, there is no great difference between the performance of the two receiving transport layer congestion management techniques. This is because there are so many ESs sending packets that the optimal window size for each ES is a low value. The simulation was therefore re-run with only a limited number of End Systems, and the result of this is summarised in Table 5-2.In this case, only a few ESs are sending packets so that the optimal window size for each ES is higher than for the previous example. The value for the size of the advertised window will therefore oscillate between 1 and a high value, with the Receiving Transport Layer Withdrawing Credit technique, while this is not the case with the Receiving Transport Layer Congestion Avoidance technique.

Detailed analysis of the simulations has shown that the Receiving Transport Layer Withdrawing Credit technique reacts more efficiently to a sudden increase in network load, however, the Receiving Transport Layer Congestion Avoidance technique avoids the oscillating behaviour of the Receiving Transport Layer Withdrawing Credit technique.

	mean end-to-end delay (seconds)	mean throughput (i.e. the number of successfully transferred packets per second without taking into account retransmissions)	power (i.e. throughput / end-to-end delay)	percentage of packets that are discarded due to congestion	ratio of number of successfully transferred packets to the number of transmitted packets without taking into account retransmissions	ratio of number of successfully transferred packets to the number of transmitted packets taking into account retransmissions	mean buffer load taken at a congested IS	ratio of the number of Error NPDUs sent that indicated congestion to the number of NPDUs sent (Data and Error)
No congestion management strategy	25.5822	0.491	0.0192	0.0204	0.9888	0.7979	189219	
Receiving Transport Layer Congestion Avoidance	2.8783	0.5773	0.2006	0	0.9983	0.9917	12442	
Receiving Transport Layer Withdrawing Credit	4.3851	0.559	0.1274	0	0.9979	0.9610	22877	
Sending Transport Layer Backoff	21.873	0.4975	0.0227	0	0.9930	0.8461	154676	
Sending of an Error NPDU indicating congestion	12.8315	0.4953	0.0386	0	0.9946	0.8453	86293	0.012

Table 5-2 Simulation Results with Limited Number of End Systems

It can therefore be concluded from the results summarised in Table 5-2, that the Receiving Transport Layer Congestion Avoidance technique gives better results.

In Table 5-1, the throughput is the best when the “Sending Transport Layer Backoff” algorithm is used, however the end-to-end delay and the buffer load are also the worst with that algorithm. This can be explained by the transport layer only reacting when packets get discarded due to congestion, so that the buffers will get full. When the buffers are full, the end-to-end delay automatically increases. Throughput is very high because the buffers never get empty. With the other algorithms, the buffers occasionally get empty, so that no optimal use of the network resources is made. The number of retransmissions is higher than for the other algorithms, but does not lower the throughput to a value lower than for the other congestion management techniques.

Notes on the results for each algorithm are given below.

5.1 Receiving Transport Layer Congestion Avoidance

This technique has far better results if a very small value is taken for the parameter par_{ce}. The best results were obtained if the buffer load was kept very small, i.e. if the congestion experienced flag is set very quickly.

The values 0.500, 0.625, 0.750 and 0.875 were taken for $par_{decrease}$. The best results were obtained for the values 0.500 and 0.625. If the size of the advertised window was only decreased slightly (i.e. by multiplying the size by 0.875) congestion couldn't be avoided.

The initial window size should be very small (a value of 1 or 2).

5.2 Receiving Transport Layer Withdrawing Credit

As with the previous exercise it appears to be important that the congestion experienced flag is set very quickly.

The values 1, 2, 3 and 4 were taken for par_{mult}. The results were comparable.

5.3 Sending Transport Layer Backoff

The value of the initial threshold is of no great importance.

As the level of congestion increases, a significant increase in the end-to-end delay and in the buffer load can be noticed, but throughput does not suffer.

5.4 Sending of an Error NPDU indicating congestion

As the value for par_{ce} increases, the throughput increases but the end-to-end delay and the buffer load also increase.

As the network load is high, the overhead that is caused by the sending of the Error NPDUs indicating congestion, cannot be ignored.

In Table 5-1par_{ce} equalled 50% of the queue capacity.

ANNEX A: VALIDATION EXERCISE SPECIFICATIONS

1. Approach adopted

An evolutionary approach to the simulation exercises has been developed. First simple validation exercises have been executed. These exercises examined only one aspect of the congestion management. With these exercises values for certain parameters of the techniques under examination were determined. The values for these parameters were then used in more complicated exercises.

The following exercises were performed:

- Exercise No Congestion Management Strategy
- Exercise Receiving Transport Layer Congestion Avoidance
- Exercise Receiving Transport Layer Withdrawing Credit
- Exercise Sending Transport Layer Backoff
- Exercise Sending of an Error NPDU indicating congestion

2. Conventions

see DED1/EAS3/STA/DOC/95-05 (Conventions in support of validation exercise specifications)

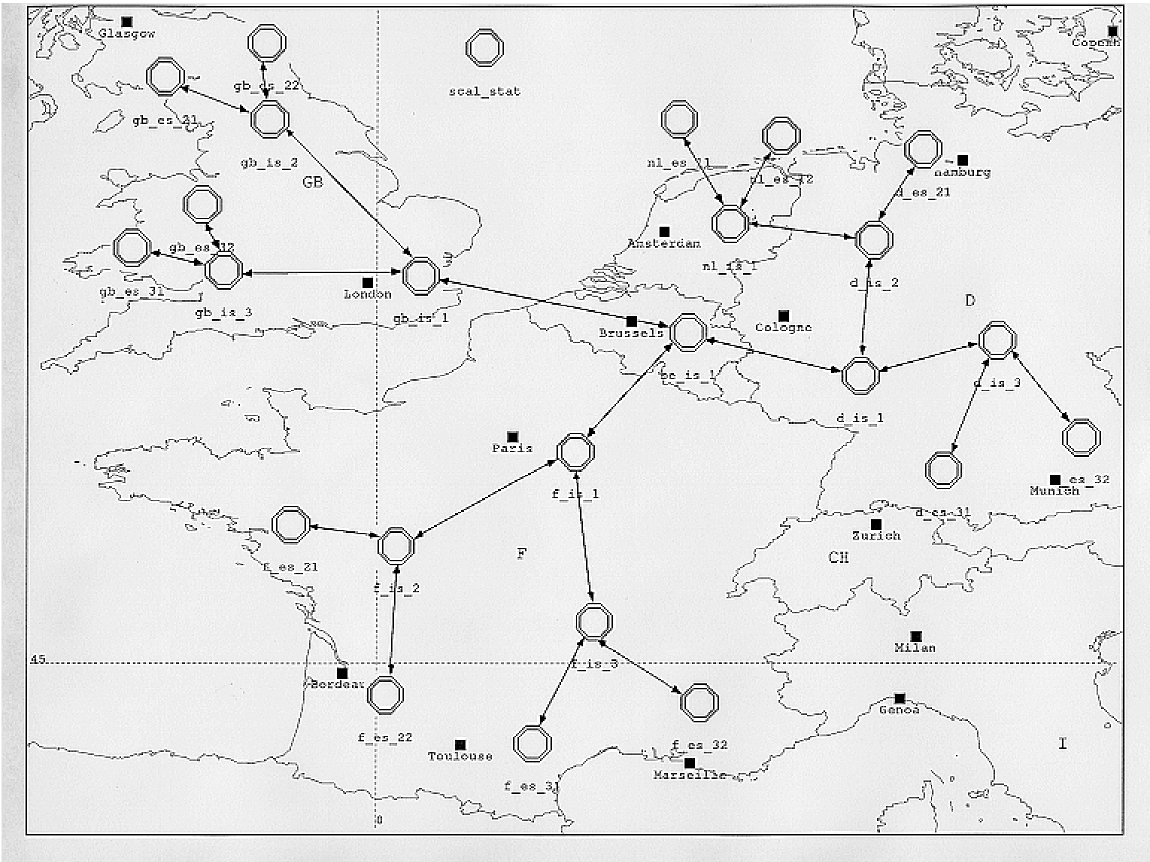
OPNET models are hierarchical. At the lowest level, the behaviour of an algorithm or protocol is encoded by a state/transition diagram with embedded code based on C language constructs. At the middle level, discrete functions such as buffering, processing, transmitting, and receiving data packets are performed by separate objects, some of which rely on an underlying process model. These objects, called modules, are connected to form a higher level node model. At the highest level, node objects based on underlying node models are deployed and connected by links to form a higher level node model. At the highest level, node objects based on underlying node models are deployed and connected by links to form a network model. The network model defines the scope of a simulation, and it is used as a “table of contents” when the simulation executable is bound together from its discrete components.

In order to describe a simulation exercise for OPNET, it is necessary to describe the configuration of the exercise at the network level and at the node level. The network configuration is described according to the conventions in document “DED1/EAS3/STA/DOC/95-05 (Conventions in support of validation exercise specifications)”. The configuration of the nodes used in the configuration is described using OPNET modules.

3. Configurations

3.1 Network configurations

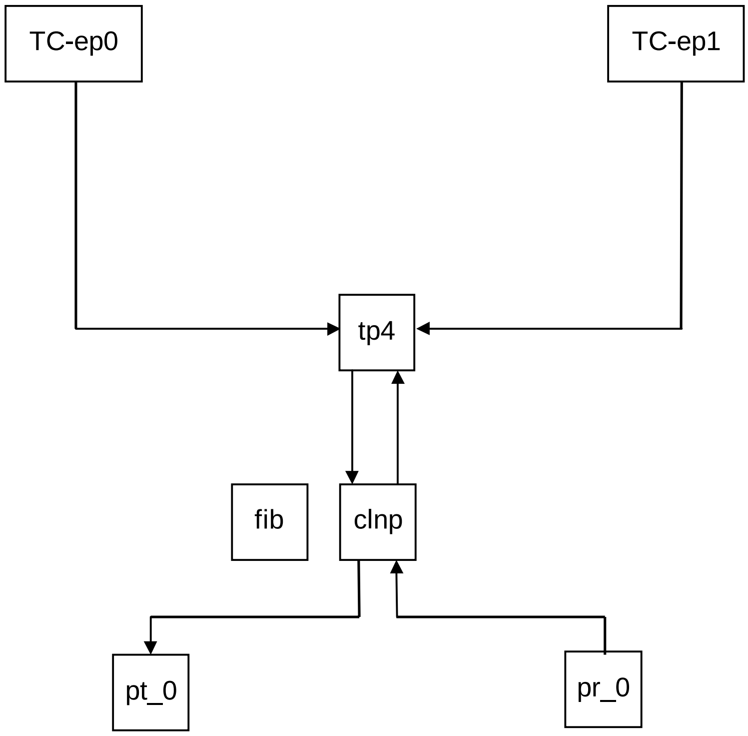
The configuration for the simulation exercises consists of 11 ISs with 13 ESs connected to them.



3.2 Node configurations

3.2.1 Host node - Integrated TP4-CLNP model

The configuration of such a node is:



with	TP-ep_*x*	a Transport Connection end point process
	fib	the forwarding information base
	tp4	process modelling the TP4 protocol
	clnp	process modelling the CLNP protocol
	pt_*x*	transmitter module
	pr_*x*	receiver module

One or more TC-ep processes are attached to the TP4 process of each host system. The TC-ep processes serve as traffic generators for the TP4 processes. They set up different transport connections to different destinations with different priorities. Once a transport connection is set up, a TC-ep process will generate packets to be sent over the transport connection. Each TC-ep process is identified by an unique TSAP selector. If the TP4 process receives packets for a particular TSAP selector, it transmits it to the corresponding TC-ep process which discards it.

A TC-ep process satisfies the following requirements:

- Transport connections are set up at times determined according to a specified probability function f_{TC}; the mean time between two successive transport connections is an attribute of the model (say $attr_{mean\_inter\_TC\_time}$);

- Transport connections are released at a time that is determined according to a specified probability function f_{TC_dur}; the mean duration of a transport connection is an attribute of the model (say $att_{mean\_TC\_duration}$);

- The peer transport entity of a transport connection is determined according to a specified uniform probability function f_{peer_TE};

- The priority of each transport connection is determined according to a specified uniform probability function f_{pri};

- When a transport connection is established:

  - Packets are generated at times determined according to a specified probability function f_{gen}; the mean time between the production of successive packets is an attribute of the generator (say $attr_{mean\_interproduction\_time}$);

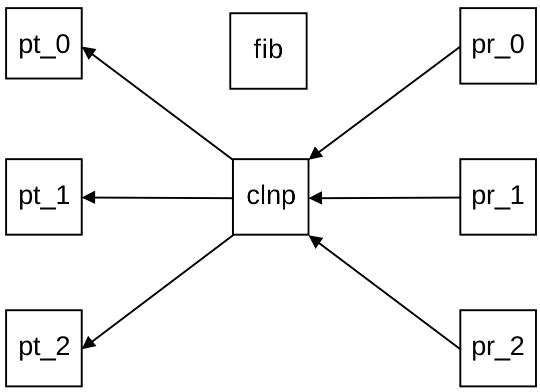
  - Packet sizes are determined according to a specified probability function f_{size}; the mean packet length is an attribute of the generator.

The fib process is just a data-store for forwarding instructions. It can be fed by routing protocols or by a static configuration file.

Different nodes are connected by communication links. Transmitter modules serve as the outbound interface between packet streams inside a node and communication links outside the node. Receiver modules serve as the inbound interface between communication links outside a node and packet streams inside the node. When a packet is given to a transmitter in a source node, the packet is conveyed over a communication link to a receiver in the destination node.

3.2.2 Router node

The configuration of such a node is:



with	fib	the forwarding information base
	clnp	process modelling the CLNP protocol
	pt_*x*	transmitter module
	pr_*x*	receiver module

The number of transmitter and receiver modules varies according to the number of communication links attached to the node.

For a description of the fib and the transmitter and receiver modules, see section “3.2.1 Host node - Integrated TP4-CLNP model”.

4. Communication profiles

This section gives the tables of the communication profiles that are needed for the ATN entities used in the exercises

4.1 CLNPis-cong-1 Profile

Index	ATN Options	CLNPis-cong-1
icong-1	Setting of the *congestion experienced* flag	yes

4.2 CLNPis-cong-2 Profile

Index	ATN Options	CLNPis-cong-2
icong-1	Sending of an error NPDU indicating congestion	yes

4.3 TP4-cong-1 Profile

Index	ATN Options	TP4-cong-1
icong-1	Receiving Transport Layer Congestion avoidance	yes

4.4 TP4-cong-2 Profile

Index	ATN Options	TP4-cong-2
icong-1	Receiving Transport Layer Withdrawing Credit	yes

4.5 TP4-cong-3 Profile

Index	ATN Options	TP4-cong-3
icong-1	Sending Transport Layer Backoff	yes

5. Specifications

5.1 Exercise No Congestion Management Strategy

5.1.1 Exercise Reference

STU/TP4/ 1

5.1.2 Objective Reference

Identification of congestion management strategy

5.1.3 Configuration

A medium size internet suffices for this exercise (see section 3.1).

A host system is modelled by the “Host node - Integrated TP4-CLNP model” (see section 3.2.1).

A router system is modelled by the “Router node” (see section 3.2.2).

5.1.4 Communication profiles

Profile for a host system: TP4-0 + CLNPes-0

Profile for a router system: CLNPis-0

5.1.5 Specification

The purpose of this exercise is to examine the behaviour of the network for increasing level of network charge when no congestion management strategy is exercised.

```
for ( mean_interproduction_time = v_i .. v_e )
{
        for ( each host h )
        {
                for ( each traffic generator of h )
                {
                        attr_mean_interproduction_time = mean_interproduction_time;
                }
        }
        run simulation;
        collect results;
}
```

The values for $attr_{mean\_interproduction\_time}$ should be small enough so that congestion is experienced.

5.1.6 Results

The simulation model records, as global statistics, the following results:

- the mean end-to-end delay;
- the mean throughput;
- the percentage of packets that are discarded due to congestion;
- the ratio of the number of successfully transferred packets to the number of transmitted packets, without taking into account retransmissions;
- the ratio of the number of successfully transferred packets to the number of transmitted packets, taking into account retransmissions;

The simulation model records, as local statistics, the following results:

- the buffer load within each IS.

5.2 Exercise Receiving Transport Layer Congestion Avoidance

5.2.1 Exercise Reference

STU/TP4/ 2

5.2.2 Objective Reference

Identification of congestion management strategy

5.2.3 Configuration

A medium size internet suffices for this exercise (see section 3.1).

A host system is modelled by the "Host node - Integrated TP4-CLNP model" (see section 3.2.1).

A router system is modelled by the "Router node" (see section 3.2.2).

5.2.4 Communication profiles

Profile for a host system: TP4-0 + TP4-cong-1 + CLNPes-0

Profile for a router system: CLNPis-0 + CLNPis-cong-1

5.2.5 Specification

The purpose of this exercise is to examine the "Receiving Transport Layer Congestion Avoidance" technique.

At the network layer one parameter needs to be considered:

- par_{ce}: When the depth of a particular output queue exceeds a certain threshold par_{ce}, the network layer will set the *congestion experienced* flag for packets in that queue.

At the transport layer one parameter needs to be considered:

- $par_{decrease}$: If the size of the advertised window needs to be decreased, it should be decreased by multiplying the size by $par_{decrease}$.

These parameters are attributes of the model.

The intention of this exercise is to find out what are proper values for par_{ce} and $par_{decrease}$.

```
for ( mean_interproduction_time = v_i .. v_e )
{
        for ( par_ce = par_ce i .. par_ce e )
        {
                for ( par_decrease = par_decrease i.. par_decrease e )
                {
                        for ( each host h )
                        {
                                for ( each traffic generator of h )
                                {
                                        attr_mean_interprod_time = mean_interprod_time;
                                }
```

```
                attr_decrease = par_decrease;
            }
            for ( each router r )
            {
                attr_ce = par_ce;
            }
            run simulation;
            collect results;
        }
    }
}
```

The values for $attr_{mean\_interproduction\_time}$ should be small enough so that congestion is experienced.

5.2.6 Results

The simulation model records, as global statistics, the following results:

- the mean end-to-end delay;
- the mean throughput;
- the percentage of packets that are discarded due to congestion;
- the ratio of the number of successfully transferred packets to the number of transmitted packets, without taking into account retransmissions;
- the ratio of the number of successfully transferred packets to the number of transmitted packets, taking into account retransmissions;

The simulation model records, as local statistics, the following results:

- the buffer load within each IS.

The results for each simulation should be compared with one another so that appropriate values for par_{ce} and $par_{decrease}$ can be determined.

5.3 Exercise Receiving Transport Layer Withdrawing Credit

5.3.1 Exercise Reference

STU/TP4/ 3

5.3.2 Objective Reference

Identification of congestion management strategy

5.3.3 Configuration

A medium size internet suffices for this exercise (see section 3.1).

A host system is modelled by the “Host node - Integrated TP4-CLNP model” (see section 3.2.1).

A router system is modelled by the “Router node” (see section 3.2.2).

5.3.4 Communication profiles

Profile for a host system: TP4-0 + TP4-cong-2 + CLNPes-0

Profile for a router system: CLNPis-0 + CLNPis-cong-1

5.3.5 Specification

The purpose of this exercise is to examine the "Receiving Transport Layer Withdrawing Credit" congestion management technique.

At the network layer one parameter needs to be considered:

- par_{ce}: When the depth of a particular output queue exceeds a certain threshold par_{ce}, the network layer will set the *congestion experienced* flag for packets in that queue.

At the transport layer one parameter needs to be considered:

- par_{mult}: When congestion has been experienced, the advertised window is decreased. Afterwards the window may be increased by one each time n packets arrive where n equals "par_{mult} x size(advertised window)".

These parameters are attributes of the model.

The intention of this exercise is to find out what are proper values for par_{ce} and par_{mult}.

```
for ( mean_interproduction_time = v_i .. v_e )
{
        for ( par_ce = par_ce i .. par_ce e )
        {
                for ( par_mult = par_mult i.. par_mult e )
                {
                        for ( each host h )
                        {
                                for ( each traffic generator of h )
                                {
                                        attr_mean_interproduction_time = mean_interproduction_time;
                                }
                                attr_sampling-period = par_sampling-period;
                                attr_mult = par_mult;
                        }
                        for ( each router r )
                        {
                                attr_ce = par_ce;
                        }
                        run simulation;
                        collect results;
                }
        }
}
```

The values for $attr_{mean\_interproduction\_time}$ should be small enough so that congestion is experienced.

5.3.6 Results

The simulation model records, as global statistics, the following results:

- the mean end-to-end delay;
- the mean throughput;
- the percentage of packets that are discarded due to congestion;
- the ratio of the number of successfully transferred packets to the number of transmitted packets, without taking into account retransmissions;
- the ratio of the number of successfully transferred packets to the number of transmitted packets, taking into account retransmissions;

The simulation model records, as local statistics, the following results:

- the buffer load within each IS.

The results for each simulation should be compared with one another so that appropriate values for par_{ce} and par_{mult} can be determined.

5.4 Exercise Sending Transport Layer Backoff

5.4.1 Exercise Reference

STU/TP4/ 4

5.4.2 Objective Reference

Identification of congestion management strategy

5.4.3 Configuration

A medium size internet suffices for this exercise (see section 3.1).

A host system is modelled by the "Host node - Integrated TP4-CLNP model" (see section 3.2.1).

A router system is modelled by the "Router node" (see section 3.2.2).

5.4.4 Communication profiles

Profile for a host system: TP4-0 + TP4-cong-3 + CLNPes-0

Profile for a router system: CLNPis-0

5.4.5 Specification

The purpose of this and the next exercise is to examine the "Sending Transport Layer Backoff" congestion management technique.

In this exercise only one aspect of this strategy is examined, namely the adjustment of the congestion window. Only one parameter needs to be determined, namely $par_{initial\ threshold}$ which indicates the initial value of the threshold. This parameter are an attribute of the model.

The strategy is only applied when a packet needs to be retransmitted because the retransmission timer expired, and not when a N-report primitive indicating congestion was received from the network layer (the model that is used for the network layer doesn't generate these reports).

```
for ( mean_interproduction_time = vi .. ve )
{
```

```
        for ( par_initial threshold = par_initial threshold i .. par_initial threshold e )
        {
                for ( each host h )
                {
                        for ( each traffic generator of h )
                        {
                                attr_mean_interproduction_time = mean_interproduction_time;
                        }
                        attr_initial threshold = par_initial threshold;
                }
                run simulation;
                collect results;
        }
}
```

The values for $attr_{mean\_interproduction\_time}$ should be small enough so that congestion is experienced.

5.4.6 Results

The simulation model records, as global statistics, the following results:

- the mean end-to-end delay;
- the mean throughput;
- the percentage of packets that are discarded due to congestion;
- the ratio of the number of successfully transferred packets to the number of transmitted packets, without taking into account retransmissions;
- the ratio of the number of successfully transferred packets to the number of transmitted packets, taking into account retransmissions;

The simulation model records, as local statistics, the following results:

- the buffer load within each IS.

The results for each simulation should be compared with one another so that appropriate values for $par_{initial\ threshold}$ can be determined.

5.5 Exercise Sending of an Error NPDU indicating congestion

5.5.1 Exercise Reference

STU/CLNP/ 1

5.5.2 Objective Reference

Identification of congestion management strategy

5.5.3 Configuration

A medium size internet suffices for this exercise (see section 3.1).

A host system is modelled by the “Host node - Integrated TP4-CLNP model” (see section 3.2.1).

A router system is modelled by the “Router node” (see section 3.2.2).

5.5.4 Communication profiles

Profile for a host system: TP4-0 + TP4-cong-3 + CLNPes-0

Profile for a router system: CLNPis-0 + CLNPis-cong-2

5.5.5 Specification

In this exercise the two aspects of the “Sending Transport Layer Backoff” congestion management technique are examined, i.e. the adjustment of the congestion window at the transport layer and the sending of an Error NPDU at the network layer. So the “Sending Transport Layer Backoff” strategy is activated if a packet needs to be retransmitted because the retransmission timer expired or when a N-REPORT primitive indicating congestion was received from the network layer (only the former was taken into consideration in the previous exercise).

For the “Sending of an error NPDU indication congestion”, the following parameter needs to be considered:

- $par_{error\ NPDU}$: When the depth of a particular output queue exceeds a certain threshold $par_{error\ NPDU}$, the network layer will send an error NPDU indicating congestion to the sender of the Data NPDU.

As a result of the previous exercise an appropriate value for the parameter $par_{initial\ threshold}$ should have been determined, so that this parameter can be constant in this exercise.

The purpose of this exercise is to find out proper values for this parameter and to examine the overhead introduced by the error NPDUs that are sent by the network layer.

```
for ( mean_interproduction_time = v_i .. v_e )
{
        for (par_error NPDU = par_error NPDU i .. par_error NPDU e )
        {
                for ( each host h )
                {
                        for ( each traffic generator of h )
                        {
                                attr_mean_interproduction_time = mean_interproduction_time;
                        }
                }
                for ( each router r )
                {
                        attr_error NPDU = par_error NPDU;
                }
                run simulation;
                collect results;
        }
}
```

The values for $attr_{mean\_interproduction\_time}$ should be small enough so that congestion is experienced.

5.5.6 Results

The simulation model records, as global statistics, the following results:

- the mean end-to-end delay;

- the mean throughput;
- the percentage of packets that are discarded due to congestion;
- the ratio of the number of successfully transferred packets to the number of transmitted packets, without taking into account retransmissions;
- the ratio of the number of successfully transferred packets to the number of transmitted packets, taking into account retransmissions;
- the ratio of the number of Error NPDUs sent that indicated congestion to the number of NPDUs sent (Data and Error)

The simulation model records, as local statistics, the following results:

- the buffer load within each IS.

The results for each simulation should be compared with one another so that appropriate values for $par_{error\ NPDU}$ can be determined. The overhead introduced by the sending of the error NPDUs should also be examined.

The results of the exercises “5.1 Exercise No Congestion Management Strategy”, “5.2 Exercise Receiving Transport Layer Congestion Avoidance”, “5.3 Exercise Receiving Transport Layer Withdrawing Credit”, “5.4 Exercise Sending Transport Layer Backoff” and “5.5 Exercise Sending of an Error NPDU indicating congestion” should be compared with one another.