XTP and Multimedia ?

Thomas Schütt, Jochen Sandvoss, Ralf Steinmetz

IBM European Networking Center, Heidelberg, Germany

[SSSt93] *Jochen Sandvoss, Thomas Schütt, Ralf Steinmetz;* **XTP and Multimedia?**; IEEE Global Telecommunication Conference (Globecom '93), Houston, Texas, 29. November - 2. December 1993, erweiterte englische Fassung von [SSSS93].

Abstract

This paper analyses the requirements of multimedia communications in terms of XTP (Xpress Transfer Protocol). In a first step, multimedia communication requirements are described. Subsequently individual features of XTP relevant for audio and video data transfer are presented. We describe the multimedia requirements and provide a possible way to satisfy them using XTP. It turns out that a multimedia capable XTP requires new interpretations of xisting mechanisms as well as some extensions.

1. Introduction

Integrated multimedia systems provide the possibility for system supported generation, processing, presentation, storage and communication of independent discrete media like text or graphic as well as continuous media like audio or video [16]. Local multimedia systems are already available as products. Distributed multimedia systems form the basis for many interesting applications. Besides multimedia dialog systems also distribution services are evolving. Other applications are for instance multimedia conferencing [2] or an individual newspaper [6]. These applications depend as a necessary precondition on a communication subsystem, that suits to the requirements of multimedia data.

The ISO transport protocols could be extended in order to support those systems with the capability to handle multimedia traffic [19]. After some modifications also TCP can be used to transmit continuous data [3]. However those protocols are not suited per se to the properties of audio and video - they were extended afterwards. Other interesting protocol ideas are NetBLT (Network Block Transfer Protocol) from the MIT [11], VMTP (Versatile Message Transfer Protocol) from Stanford University [10], TP++ from Bellcore [14] and XTP (Xpress Transfer Protocol) [23].

In the HeiTS project at the European Networking Center of IBM we currently use two different network layer protocols: a modified X.25 protocol as well as the *Internet Stream Protocol ST-II* defined in the Internet community [21]. Implementations are available for OS/2 and AIX platforms. The experiences made with the extended X.25 protocol helped us to derive proposals for possible extensions of the ST-II protocol.

In this context XTP was analyzed as a possible transport protocol. In the following paper we summarize our experiences on this approach.

In section 3 we give a short overview on the XTP protocol based on the requirements of distributed multimedia applications. In chapter 4 to 8 we will present a more detailed view on certain protocol mechanisms. Firstly we explain the mechanism currently used in XTP. Secondly we apply those mechanisms in the context of the transmission of continuous media and discuss possible alternatives.

2. Requirements of multimedia applications

The CCITT has defined a classification scheme for multimedia applications in its recommendation I.211 [7]. According to this commonly accepted reference scheme we can distinguish between "interactive services" and "distribution services". These two classes of applications impose significantly different requirements upon a transport system used to convey such information (see also [4; 15; 17; 22]).

Taking a closer look at the **interactive services**, a point to point connection can be used to transfer data, graphics, audio or video between the cooperating application entities. Details of their respective requirements are given below:

1. **Data:** Typically, and this is a major difference to other types of information, correctness of the data down to the bit level is a must for data communication. Bandwidth requirements and end-to-end delay depend on the data volume and the application context [20].
2. **Still pictures and graphics:** For the transmission of uncompressed still images the correctness of all bits is typically not essential. For certain special applications, e.g. X-ray pictures in medicine, every single pixel is essential and must be transmitted correctly. Depending on the compression scheme, the significance of individual bits and bytes may vary. Similar to the

pure data scenario the bandwidth requirement and the maximum acceptable end-to-end delay are highly application dependent.

3. **Video:** For video transmission we have to distinguish between compressed and uncompressed representations. Reliability requirements for uncompressed video transmission are lower than for compressed transmissions. Using uncompressed representation, loss of an individual TPDU is usually acceptable because the next picture will overwrite the error within a fraction of a second. This is different for compressed video sequences. Most compression algorithms, e.g. MPEG-1 or DVI-PLV, are based on the exploitation of redundancy between consecutive pictures. Throughput requirements are in the order of 1 to 2 Mbps for compressed video transmission and up to 140 Mbps for uncompressed information, for HDTV quality even higher. Because moving videos, representing time critical continuous data, are here considered as an interactive service, a maximum end-to-end delay of 600 msec must be guaranteed. More adequate are values upto 200 msec [20]. Also the delay variation, "jitter", must be considered and may have a significant effect in particular on the memory requirements.
4. **Audio:** Restricting the view to the interactive service perspective, we have the same end-to-end delay requirements as for video. However the memory requirements to smooth out delay variations are typically one order of magnitude less than for video. The required throughput without compression depends on the audio quality and ranges from 64 kbps (telephone) up to 173 kBps (stereo in CD quality).
5. **Mixed media:** Quite frequently there are several separate but related information streams for data, still pictures, audio and video maintained between communication partners (i.e. transport service users). To ease the application design, the communication subsystem could provide support for the common management of such mutually dependent communication relations.

To emphasize: One of the consequences of the interactive communication relation is that end-to-end delay is not only critical for audio and video, but can also be absolutely essential for other data types. E.g. the data to be transferred could represent a mouse pointer which should point to a specific detail of a graph simultaneously to a spoken comment.

In a second example let us discuss multimedia **distribution services** (point-to-multipoint). In this case the data should be delivered to multiple recipients as simultaneously as possible. Compared to the interactive services the following differences and special requirements can be identified:

- The transmission of information to the different recipients should not be achieved by sending multiple copies to the individual recipients, because the effort of the sender and the burden on the communication system increases proportionally with the number of recipients. The transport system should support multicasting so that a single TPDU can reach as many recipients as possible.
- For *multicast* distribution lists need to be defined and administered. This includes all operations which effect the status of a group, e.g. creating or deleting multicast groups, entering or leaving of a group, functions related to error recognition and -correction. The transport system should provide the corresponding internal support functions or make use of an external directory service.
- The transport system should allow its user to specify a liability class according to the required reply semantics (see [18]).
- The maximum acceptable end-to-end delay is again application dependent. However typically higher values up to 1 second are acceptable.

3. Xpress Transfer Protocol XTP

XTP [23] is a light-weight, real-time transfer protocol and was designed at *Protocol Engines Incorporated* as part of the *Protocol Engine Projects* [9]. XTP is designed to run on top of modern high-speed networks. Therefore one of the most important design goals of XTP was to allow for an easy and efficient implementation in hardware. XTP uses flexible, leightweigth protocol mechanisms which may be switched on and off depending on the applications needs, in order to minimize system requirements for protocol execution.

In XTP the transport layer and network layer are unified in a so called "transfer layer" [8]. This avoids introduction of redundant functionality in both layers and permits the definition of an efficient interface between the layers.

Currently there is a discussion within the ISO whether the OSI transport protocols should be complemeted by a high speed transport protocol HSTP. The protocol core of HSTP is based on the XTP protocol. As already mentioned XTP combines layer 3 and 4. The combination of two layers - transport layer and network layer - into one might be difficult to apply in certain cases like wide area networks (WANs) because the functionality of the network layer is usually provided by the network providers. Therfore, the ISO standardisation group devided XTP up into network layer functions and a transport layer protocol HSTP.

In the following sections we will follow this approach and focus on transport protocol mechanisms with end-to-end significance.

4. TPDU-format

In addition to the regular user data XTP DATA TPDUs may also include protocol control information and, as a special XTP feature, a logically separate second fixed length user data field (8 byte). This second data field ("b-tag") is intended to be used by the transport service user to transparently convey application specific control information.

Looking at the TPDU specification from the multimedia perspective, several special phenomina related to the mixture of network and transport semantics plus potential applications for the additional (control) data field need to be discussed.

One of the potential uses of the *btag*-field for the multimedia applications under consideration is to use it to synchronize several related connections. Another potential use is to transfer time stamps associated with the application data. Such timestamps could be used by the transport user to resynchronize timecritical received data or to discard information that arrives too late.

A problem with this approach in respect to the requirements given in chapter 2 is the responsibility for the supervision and enforcement of the delay and jitter limits. This is clearly a transport system and not a transport user responsibility. Guaranteed bounds for the end-to-end delay can only be supported in the network layer, because only this layer has the necessary information on the status and performance of the individual subnetworks to be traversed. Obviously the route selection has a major effect on the delay. If network internal ressources have to be reserved to guarantee certain performance properties this can also only be done within the network layer.

Besides the end-to-end delay, the delay variation (jitter) is also of prime importance. To avoid unmanagably large values, at least some hop-by-hop smoothing might be required.

If jitter compensation should be done in layer 4, the XTP protocol were to be functionally extended [1]. Alternatively the TPDU format could be extended by a new field or an already existing one could be reused. Candidates for reuse are the already mentioned btag field, which would be unavailable to the user, or the *sort* field which is currently unused if the sort flag is reset. Additionally the local XTP entities were to be extended by a buffer management system in combination with a suitable timer mechanism. As mentioned above, this could also be done in layer 3. However there are arguments to do so in the end systems:

The buffer required for jitter compensation in end systems grows proportionally with the number of transport connections needing jitter smoothing. If the jitter compensation is done in layer 3, the per hop memory requirement is significantly less than would be needed in the end system, because the delay variations to be compensated are much less. However there are typically many connections running through a gateway (hundreds), which may overcompensate the savings due to the smaller variations.

5. Priorities

XTP provides a priority based discrimination mechanism (*priority scheduling*) to control intraprotocol scheduling.

The SORT field in the TPDU headers provides a means by which TPDUs are prioritized so that some TPDUs receive preferential treatment. The SORT field is interpreted only for TPDUs with the SORT-bit (*cmd* field in the header) set.

In the context of multimedia communications at least two data classes have to be distinguished for processing:

1. **Data with real time constraints:** processing according to a real-time scheduling algorithm like *earliest deadline first* or *rate monotonic.*
2. **Data without real time constraints:** These data can be handled by using processing capacity available after processing data with real-time constraints. In this context a priority-based scheme with help of the *sort* field can be used.

In XTP the determination of the priorities for different streams is performed locally at the sender entity.

A mapping between the required quality of service parameters and the priority values is only possible if the netload for all subnets participating in the communication process is known. The right place for a priority adjustment function is the network layer where information about all subnets are available. XTP offers no such functions, even if the complete set of functions (including layer 3 and 4) is considered. The use of priority-based scheduling mechanisms to guarantee the real-time needs of multimedia applications is not possible within XTP.

6. Error Handling

If reliability requirements for the transport service user are not already fulfilled by the lower layer services, the transport layer has to provide mechanisms for error detection, notification and correction.

Error Detection
Bit errors are detected in XTP by help of two different checksums. The header checksum is mandatory, while the calculation of the checksum over the user data segement can be turned off by setting the NOCHECK bit in the *cmd* field of the header.
The detection of TPDU losts, duplicates and sequence errors is done by help of sequence numbers. The mechanisms are byte based. The sequence control mechanisms can be switched off by setting the NOERR bit in the *cmd* field of the header.

Error Notification
Mechanisms for error notification enable the receiver to inform the sender about detected errors. XTP provides two different schemes; *go-back-n* and *selective reject*.
Using the *go-back-n* mechanism, the receiver informs the sender only about the the first sequence number of lost or missing data.
More detailed information is transferred when using the *selective reject* scheme. Here the sender gets precise information about *gaps* in the data stream that were detected by the receiver.

Error Correction
The only way for error correction in XTP is by retransmission. The mechanism is based on information about lost or corrupted data passed back from the receiver entity. If the receiver is not capable to store out of sequence data, XTP offers *go-back-n* protocol.
Using the mechanisms described above, XTP is in some way suitable for multimedia applications. Because of its high flexibility, all the reliability classes mentioned in chapter 2 can be supported.

However, for time-critical data streams error correction by retransmission might not be applicable. The maximum end-to-end delay the user can accept might be exceeded in case of an error. Besides that multicasting plays an important role in distributed multimedia applications. Especially in this case retransmissions can easily cause unnecessary waste of bandwidth[1] . The mechanisms for mapping guaranteed QOS values at the transport service interface on the corresponding values of the lower layer service interfaces become also a lot more complicated if retransmissions have to be taken into account.

Two different strategies could help solving these problems. One technique is **error concealment**, which minimizes the number of retransmissions as far as possible. One way to provide error concealment is to extend the service interface in a way that allows the user to specify its reliability requirements more precisely. There are many multimedia applications where error indication is sufficient. It should be possible to devide a transport connection into logical substreams of different *importance*. In case of congestion TPDUs of substreams with lower importance are discarded first. Proposals for such scaling mechanisms can be found in [13].

The second strategy is the introduction of a **forward error correction** method. Different methods can be applied here depending on the expected error characteristics (e.g. bit errors, TPDU loss rates, etc.). Examples include the Cross Interleaved Reed Solomon Code used in CD-DA Technology or suggestions made by [5] for ATM nets.

Another error class in the case of time critical data are late TPDUs. The detection of these errors should be realized by a suitable monitoring function. Error handling strategies are provision of an indication to the transport service user with or without passing late data to the application.

7. Flow Control

An important issue in high-speed networking is flow control, because the receiver entity or any network node runs higher risk to become congested by the high transfer rate of the sender entity.

XTP provides two different mechanisms for flow control complementing each other. One mechanism is the well-known **window-based** flow control. The receiver controls the rate of the sender passing explicit sending permissions containing sequence numbers, based on the currently accessible buffer space at the receiver side.

The second mechanism is called **rate control**. Rate control is independent of the buffer space at the receiver side, but is driven by the processing power of the receiver entity [byte/s].

Because the required bandwidth as well as the temporal behavior of distributed multimedia applications are usually predictable, window-based flow control mechanisms are not suitable for these work load characteristics but only rate controlled schemes. Therefore, concerning flow control XTP is a suitable scheme for multimedia traffic, as its window-based flow control can be switched off (NOFLOW) when required and a static rate control mechanism is internally available.

For distributed multimedia applications working on top of connections with guaranteed QOS values, there should also be a rate enforcement function available at the transport

---

1 If the loss of a TPDU is detected by the sender entity, this TPDU has to be retransmitted either over the existing multipoint connection, which means to all receiver entities, or dedicated connections for retransmissions have to be provided.

service access point (TSAP), in order to ensure that the QOS values negotiated beforehand will really be observed by the transport service user.

8. Group Communications

XTP defines a *multicast*-mode, where a sender entity transmits TPDUs to a group of receiver entities in parallel (unidirectional one-to-many).

In XTP, there is no big difference between protocol mechanisms used in multicast mode and those used in point-to-point mode. The sender entity transmits a FIRST-TPDU followed by DATA-TPDUs. Error handling is provided. For possible retransmissions *go-back-n* is used. There is no selective retransmission provided. Data transmission speed is driven by the processing speed of the slowest receiver entity joining the group. In case of gaps in the data stream, the concerned receiver entity will generate a *reject packet*. This control TPDU will be send to all entities participating the group - including all receivers - to inform hem about the detected error. It would be hard to avoid congestion in case that all receiver entities detecting an error would send control TPDUs (*reject packets*) to the sender entity. In order to reduce the number of control TPDUs necessary, XTP uses a technique (*damping*) of having a receiver refrain from transmitting its control TPDUs if it receives some other control TPDUs that would render the locally generated control TPDU superfluous. A receiver allows its control TPDU to be damed if all of the control information being tracked at the transmitter is duplicated in the control TPDU of another receiver.

Because XTP doesn't provide any mechanisms for the establishment and management of multicast groups, only two different reliability classes can be provided: [12]:

reliability class k = 0, the multicast data transfer was successful, even when no receiver entity has received a TPDU correctly.

reliability class k = 1, the multicast data transfer was successful, if at least one receiver entity has received the TPDU correctly.

The multicast service provided by XTP is therefore suitable for either pure broadcast (e.g. video distribution service, k = 0) or for the search of certain information in a distributed system (e.g. search for system resources, k = 1).

An extension of the reliability classes from k = 2 to k = size of group is not possible without some changes of the protocol [18]. Certainly, XTP contains mechanisms that allow a receiver entity to join an existing multicast connection or to leave the connection, but they are not managed. Additionally mechanisms are needed at the sender entity that provide association of control TPDUs and receiver entities. The damping scheme can not be used for this type of multicast connections. The sender has to be informed about all changes to the receiver group.

Offering only unidirectional one-to-many multipoint connections, XTP makes it quite hard to implement application protocols on top of it. It is considered a drawback of the current XTP protocol to offer only unidirectional service, leaving the sending of any user control information back an open issue. XTP should therefore be modified concerning a bidirectional one-to-many service.

9. Summary and Future Work

According to its protocol mechanisms analyzed in this paper, XTP can only partially satisfy the requirements of the transport of continuous data streams. Problem areas are for example the missing quality of service parameters and, especially in the case of multipoint connections the error handling strategies. Concerning multicasting, a group management or a suitable interface to an external directory service are desirable and, from a certain group size upwards, necessary extensions.

However, with the protocol extensions suggested in this paper, multimedia communication with XTP should be feasible. Because a connection oriented network service with global reservation mechanisms for system resources of the net can not be provided by the current network layer functionality of XTP, we decided to use the ST-II protocol in the network layer. An implementation of a suitably extended XTP protocol purely used to provide transport layer functionality on top of an existing ST-II implementation is currently under development at the ENC.

References

[1] *G. Anastasi, M. Conti, E.Gregori;* **TPR: A Transport Protocol for Real Time Services in an FDDI Environment;** In: M. Johnson (Hrsg.); Protocols for High-Speed Networks; Elsevier Science Publishers B.V. (North Holland), IFIP, 1990

[2] *S. R. Ahuja, J. Robert Ensor, David N. Horn;* **The Rapport Multimedia Conferencing System;** Proceedings of the Conference on Office Information Systems, Palo Alto CA, March 1988, pp. 1-8.

[3] *David P. Anderson, Ralf Guido Herrtwich;* **Internet Communication with End-to-End Performance Guarantees;** Informatik-Fachberichte, no. 293, Springer Verlag. 1991.

[4] *Heinrich Armbrüster, Hans Jörg Rothamel;* **Breitbandanwendungen und -dienste: Qualitative und quantitative Anforderungen an künftige Netze;** ntz, vol. 43, no. 3, 1990, pp. 150-159.

[5] *Ernst W. Biersack;* **Performance Evaluation of Forward Error Correction in ATM** Computer Communication Review, ACM SIGCOMM, Vol.22, No.1, January 1992.

[6] *Steward Brand;* **The Media Lab, Inventing the Future at MIT;** Viking Penguin, 1987.

[7] *International Telegraph and Telephone Consultative Committee;* **B-ISDN Service Aspects;** CCITT Study Group XVIII, Draft Recommendation I.211, Geneva, 23-25 May 1990.

[8] *CELAR (Centre d'Electronique de L'Armement);* **Military real time local area network;** Rennes, France, February 1987

[9] *Greg Chesson;* **The Protocol Engine Project;** UNIX Review, Vol.5, No.9, September 1987, pp.70-77

[10] *David R. Cheriton, Erik Nordmark;* **Experiences from VMTP: How to achieve low response time;** H. Rudin, R. Williamson (Hrsg.); Protocols for High-Speed Networks; Elsevier Science Publishers B.V. (North Holland), IFIP, 1989, pp.43-56

[11] *David D. Clark, Mark L. Lambert, Lixia Zhang;* **NETBLT: A High Throuput Transport Protocol;** Proc. SIGCOMM'87, ACM, August 1987, pp.353-359

[12] *J. Crowcroft, K. Paliwoda;* **A Multicast Transport Protocol;** Trans. SIGCOMM, ACM, August 1988, pp.247-256

[13] *Luca Delgrossi, Christian Halstrick, Dietmar Hehmann, Ralf Guido Herrtwich, Oliver Krone, Jochen Sandvoss, Carsten Vogt;* **Media Scaling for Audiovisual Communication with the Heidelberg Transport System** Proceedings of the First ACM Multimedia, August 1993, pp.99-104

[14] *David C. Feldmeier, Ernst W. Biersack;* **Comparison of Error Control Protocols for High Bandwith-Delay Product Networks** In: M. Johnson (Hrsg.); Protocols for High-Speed Networks; Elsevier Science Publishers B.V. (North Holland), IFIP, 1990

[15] *Domenico Ferrari;* **Client Requirements for Real-Time Communication Services;** International Computer Science Institute, Technical Report 90-007, Berkeley, March 1990.

[16] *Lutz Henckel, Heiner Stüttgen;* **Transportdienste in Breitbandnetzen;** E. Effelsberg, H.W. Meuer, G. Müller (Hrsg.); Proceedings zur GI/ITG Fachtagung "Kommunikation in verteilten Diensten" in Mannheim; Springer Verlag; February 1991; pp.96-111

[17] *Dietmar Hehmann, Michael Salmony, Heinrich Stüttgen;* **Transport Services for Multimedia Applications;** Proceedings of the IFIP WG 6.1/WG 6.4 Workshop on Protocols for High Speed Networks, North Holland, 1989, 303-321

[18] *Larry Hughes;* **Multicast Response Handling Taxonomy;** Computer Communications, vol. 12 No 1, February 1989, pp. 39-46.

[19] *Thomas Schütt, Manny Farber;* **The Heidelberg High Speed Transport System: First Performance Results;** In: 3rd International IFIP WG6.1/6.4 Workshop on Protocols for High-Speed Networks, Stockholm, May 13-15, 1992 pp. 35-50.

[20] *Ralf Steinmetz, Thomas Meyer;* **Modelling Distributed Multimedia Applications;** IEEE Int. WS on Advanced Communications and Applications for HS Networks, München, March 1992.

[21] *Claudio Topolcic;* **Experimental Internet Stream Protocol, Version 2 (ST-II);** RFC 1190, October 1990.

[22] *David J. Wright, Michael To;* **Telecommunication Applications of the 1990s and their Transport Requirements;** IEEE Network Magazine, vol.4, no.2, March 1990, pp.34-40.

[23] *Protocol Engines Incorporated;* **XTP Protocol Definition Revision 3.6;** PEI 92-10, Protocol Engines Incorporated, Santa Barbara, CA 93101, January 92

IN

IEEE Global Telecommunications Conference

"COMMUNICATIONS FOR A CHANGING WORLD"

Houston, Texas • November 29 – December 2, 1993

THE WESTIN GALLERIA HOTEL

Chairman

XTP und Multimedia ?

Markus Steffens

Johann Wolgang Goethe-University of Frankfurt
Fachbereich Informatik, Institut für Telematik
Robert-Mayer-Strasse 11-15,
6000 Frankfurt

Jochen Sandvoss, Thomas Schütt, Ralf Steinmetz

IBM European Networking Center,
Tiergartenstr. 8, Postfach 10 30 68,
6900 Heidelberg

Zusammenfassung

Dieser Beitrag untersucht die Eignung der in XTP (Xpress Transfer Protocol) eingesetzten Protokollfunktionalitäten zur Kommunikation von Audio- und Videodaten. Ausgehend von den Anforderungen multimedialer Anwendungen werden die vorhandenen XTP-Mechanismen kurz erläutert. Anschließend werden für die wichtigsten Teilaspekte des Kommunikationsprotokolls (Verbindungsmanagement, Prioritätssteuerung,Flußsteuerung etc.) jeweils im Einzelnen die Anforderung von Seiten der Übertragung kontinuierlicher Medien und mögliche Lösungen im Kontext von XTP diskutiert. Es zeigt sich, daß zur Unterstützung multimedialer Kommunikationsformen sowohl geeignete Interpretationen vorhandener XTP-Protokollmechanismen wie auch Protokollerweiterungen erforderlich sind.

Abstract

This paper analyzes the requirements of multimedia communications in terms of XTP (Xpress Transfer Protocol). In a first step, multimedia communication requirements are described. Subsequently individual features of XTP relevant for audio and video data transfer are presented. We describe the multimedia requirements and provide a possible way to satisfy them using XTP. It turns out that a multimedia capable XTP requires new interpretations of existing mechanisms as well as some extensions.

1. Einleitung

Integrierte Multimedia-Systeme bieten Möglichkeiten zur rechnergestützten Erzeugung, Verarbeitung, Darstellung, Speicherung und Kommunikation unabhängiger diskreter Medien

wie Text oder Graphik sowie kontinuierlicher Medien wie Ton oder Bewegtbild [23]. Solche lokalen Systeme sind heute als Produkte auf dem Markt. Verteilte Multimedia Systeme sind die Basis für eine Menge interessanter Anwendungen. Neben Multimedia-Dialogsystemen entstehen hier auch Verteildienste, Multimedia-Konferenzsysteme [2], oder beispielsweise eine individuelle Zeitung [7]. Diese Anwendungen erfordern als wesentliches Merkmal ein Kommunikationssystem, daß den Anforderungen für Multimedia-Daten genügt.

Das ISO Transportprotokoll kann zur Unterstützung eines solchen multimediafähigen Systems erweitert werden [30]. Auch TCP kann mit einigen Veränderungen kontinuierliche Medien übertragen [3]. Jedoch sind diese Protokolle nicht per se den Eigenschaften von Audio und Video angepaßt, sie wurden nachträglich erweitert. Weitere interessante Entwicklungen sind NetBLT (Network Block Transfer Protocol) vom MIT [16], VMTP (Versatile Message Transfer Protocol) aus der Stanford University [15], TP++ von Bellcore [20] und XTP (Xpress Transfer Protocol) [14; 29; 37]. XTP bietet dabei die Möglichkeit einer Zusammenfassung der Vermittlungs- und Transportschicht.

Im HeiTS-Projekt am Europäischen Zentrum für Netzwerkforschung der IBM verwenden wir zur Zeit zwei alternative Vermittlungsprotokolle: ein modifiziertes X.25 Protokoll sowie das im amerikanischen DoD Kontext definierte *Internet Stream Protocol* ST-II. Die Implementierung erfolgte auf OS/2 bzw. AIX Plattformen. In der nächsten Stufe wird ST-II [33] in das OS/2-System integriert. Aus den mit dem erweiterten X.25 gewonnenen Erfahrungen sollen dabei ggf. Vorschläge für Erweiterungen des ST II Protokolles abgeleitet werden. In diesem Kontext wurde XTP als ein mögliches Transportprotokoll untersucht. In diesem Bericht werden die ersten Erfahrungen hierüber zusammengefaßt.

Ausgehend von den Anforderungen multimedialer Anwendungen wird in den Kapiteln 3 und 4 ein kurzer Überblick zu XTP gegeben. Die folgenden Kapitel 5 bis 10 betrachten einzelne Aspekte des Kommunikationsgeschehens, indem jeweils zuerst der XTP-Mechanismus erläutert wird. Anschließend werden jeweils die Anforderungen an einen solchen Mechanismus aus Sicht der Übertragung kontinuierlicher Medien und eine mögliche Lösung im Kontext von XTP aufgezeigt.

2. Anforderungen multimedialer Anwendungen

Im folgenden Abschnitt werden anhand von zwei Beispielanwendungen typische Anforderungen abgeleitet, die multimediale Anwendungen an ein Kommunikationssystem stellen. Hierbei wird besonders auf die benötigte Funktionalität bezüglich der Transportschicht eingegangen (siehe auch [4; 21; 24; 36]).

Das **ersten Beispiel** einer Anwendung besteht aus **Dialogdiensten** nach [11]. Eine Punkt-zu-Punkt Kommunikationsverbindung überträgt Daten, Graphiken, Sprache und Bewegtbilder zwischen den beiden Anwendungsinstanzen. Zusammenfassend lassen sich folgende Anforderungen ableiten:

1. **Daten:** Bei der Übertragung der Daten muß die Fehlerfreiheit auf Bitebene gewährleistet sein. Die benötigte Bandbreite und Ende-zu-Ende-Verzögerung hängen vom Datenvolumen und wesentlich vom Anwendungskontext ab [32]. Ein Dateitransfer benötigt keine reservierte Bandbreite und keine garantierte Ende-zu-Ende-Verzögerung; er sollte nur möglichst schnell erfolgen. Sind diese Daten beispielsweise aber der Status und die Koordinaten eines Zeigers auf einem gemeinsamen Fenster, dann bestehen Forderungen bezüglich der maximalen Ende-zu-Ende-Verzögerung. Hieraus läßt sich zusammen mit

der charakteristischen Datenrate eine untere Schranke der benötigten Bandbreite ableiten.

2. **Einzelbilder und Graphiken:** Für die Übermittlung von unkomprimierten Einzelbildern ist die Korrektheit aller Bits im Allgemeinen nicht entscheidend. Ein verfälschter, einzelner Farbwert stört den optischen Eindruck nicht in einem Bild, das aus beispielsweise 1000 mal 1000 Bildpunkten aufgebaut ist. Geht hingegen eine ganze Zeile eines Bildes verloren, dann kann dies gegebenenfalls schon untragbar sein. Bei manchen Anwendungen, beispielsweise Röntgenaufnahmen in der Medizin, ist dagegen häufig jedoch jeder Bildpunkt relevant. Bei komprimierten Einzelbildern besitzen unterschiedliche Bits und Bytes in einem Datenstrom verschiedene Relevanz. So sind beispielsweise im JPEG-Format nach einer sequentiellen Kompression die DCT-Koeffizienten mit den niedrigsten zwei-dimensionalen Frequenzen sehr wichtig, während die DCT-Koeffizienten mit den höchsten Frequenzen relativ unwichtig sind. Der benötigte Bandbreitenbedarf sowie die maximal tolerierbare End-zu-End-Verzögerung sind wie bei Daten stark anwendungsabhängig. Für die zur Übertragung eines graphischen Objektes benötigte Bandbreite sind außerdem das gewählte Darstellungsverfahren, die Größe, Auflösung pro Pixel, Kodierung (beispielsweise 9-bit YUV) und Kompression wesentlich.

3. **Bildsequenzen:** Bei der Übermittlung von Bewegtbildern muß zwischen komprimierter und unkomprimierter Bildübertragung unterschieden werden. Die Zuverlässigkeitsanforderungen für unkomprimierte Bewegtbildübertragung sind geringer als für komprimierte Bewegtbildübertragung. Bei der unkomprimierten Bildübertragung sind TPDU-Verluste erträglich, da innerhalb von Sekundenbruchteilen das Folgebild den Fehler überlagert. Nicht so bei komprimierter Bewegtbildübertragung. Die meisten Kompressionsverfahren (wie MPEG-1, DVI-PLV-Mode) basieren auf einer Redundanzreduktion von zeitlich aufeinander folgenden Bildern. Hier folgt meist einem „interframe" komprimiertem Einzelbild (in MPEG das „I-Frame") mehrere „intraframe" komprimierte Bilder (in MPEG die „P und B-Frames"). Datenverlust und -verfälschungen wirken sind somit unterschiedlich aus. Der benötigte Durchsatz liegt bei komprimierter Bewegtbildübertragung im Bereich von 1 - 2 MBit/s, bei unkomprimierter Übertragung im Bereich bis zu 140 MBit/s (Bei HDTV-Qualität geht dies bis in den GBit/s-Bereich hinein). Da Bewegtbilder zeitkritische (kontinuierliche) Daten darstellen und es sich hier um eine Dialoganwendung handelt, ist eine maximale Übertragungsverzögerung von 600 ms unbedingt einzuhalten. Günstiger sind Werte um die 200 ms [32]. Auch die Schwankung der Übertragungsverzögerung „Jitter" ist zu beachten. Diese äußert sich insbesondere in den Speicherplatzanforderungen.

4. **Sprache:** Für Sprache gelten wegen der hier betrachteten Dialoganwendung und den zeitkritischen (kontinuierlichen) Daten die gleichen Anforderung bezüglich der maximalen Übertragungsverzögerung wie bei den Bewegtbildern. Die aus den maximalen Schwankungen der Übertragungsverzögerung abgeleiteten Speicherplatzanforderungen sind meist ca. um den Faktor 5-10 geringer als bei Bewegtbildern. Hier sind die erforderliche Qualität zusammen mit dem verwendeten Kompressionsalgorithmus die bestimmenden Faktoren. Der vom Benutzer akzeptierbare Jitter ist aber im Vergleich zur Bewegtbildübertragung geringer, weil das menschliche Ohr empfindlicher als das Auge ist. Der benötigte Durchsatz liegt ohne Kompression, in Abhängigkeit der Tonqualität, im Bereich von 64 kBit/s (Telefon) bis 173 kByte/s (Stereo in CD-Qualität).

5. **Medienmix:** Zwischen den Kommunikationspartnern (Transportdienstbenutzern) bestehen oft verschiedene Verkehrsströme für Daten, Einzelbild, Audio und Bewegtbilder.

Deshalb sollte das Kommunikationssystem eine Möglichkeit zur gemeinsamen Verwaltung solcher voneinander abhängiger Kommunikationsbeziehungen bieten.

An dieser Stelle sei nochmals explizit auf die Konsequenz der betrachteten interaktiven Kommunikationsbeziehung eingegangen: Es kommt nicht nur bei der Bewegtbild- und Sprachübertragung auf die Übertragungsverzögerung an, sondern auch bei den anderen Medien. So können beispielsweise die zu übertragenden Daten einen Maus*pointer* darstellen, der gleichzeitig, zu einem gesprochenen Kommentar, auf ein Detail in einer Graphik zeigen soll.

Im einem **zweiten Beispiel** sei ein **multimedialer Verteildienst** (Punkt-zu-Mehrpunkt) betrachtet. Hier sollen die Daten von einem Sender möglichst gleichzeitig zu verschiedenen Benutzern übertragen werden. Zusammenfassend lassen sich folgende, vom ersten Beispiel abweichende, Anforderungen ableiten:

- Die Übertragung der Daten zu den verschiedenen Benutzern sollte nicht durch mehrfaches Senden an die einzelnen Empfänger nachgebildet werden, da der Aufwand proportional mit der Anzahl der Empfänger steigt. Das Transportsystem sollte *Multicasting* unterstützen, damit mit einer einzigen TPDU möglichst viele Empfänger erreicht werden können.
- Beim *Multicast* müssen die Empfängergruppen festgelegt und verwaltet werden. Hierzu gehören alle Operationen, die den Status der Gruppe betreffen wie das Erzeugen und das Vernichten von *Multicast*-Gruppen, der Eintritt in die Gruppe und das Verlassen der Gruppe, Operationen im Zusammenhang mit Fehlererkennung und -behebung. Das Transportsystem sollte entsprechende Funktionalität enthalten oder einen extern verfügbaren *Directory*-Dienst verwenden.
- Das Transportsystem sollte dem Dienstbenutzer ermöglichen, einen Zuverlässigkeitsgrad entsprechend der Antwort-Semantik anzugeben (siehe [25]).
- Die maximale End-zu-End-Verzögerung ist wiederum anwendungsabhängig, es lassen sich jedoch meist höhere Werte im Bereich von maximal 1 Sek. tolerieren.

Klasse	Bitfehler	TSDU-Fehler
0	keine Sicherung	keine Sicherung
1	keine Sicherung	Sicherung mit Anzeige
2	Sicherung mit Anzeige	Sicherung mit Anzeige
3	keine Sicherung	Sicherung mit Behebung
4	Sicherung mit Behebung	Sicherung mit Behebung

Tabelle 1. Zuverlässigkeitsklassen zur Fehlerbehebung

Über die folgenden Dienste und Dienstgüte-Parameter können die Anforderungen an ein Transportsystem aus Sicht der multimedialen Anwendung befriedigt werden:
Neben der Punkt-zu-Punkt-Kommunikation ist eine Gruppen-Kommunikation notwendig. Der Dienstgüte-Parameter „Durchsatz" kann über die TSDU-Rate (TSDUs/s) zusammen mit einer mittleren Datenrate (Bytes/s) und maximalen TSDU-Größe (Bytes/TSDU) spezifi-

ziert werden. Die maximale Übertragungsverzögerung wird mit dem *delay*-Parameter (ms) angegeben. Die Schwankung der Übertragungsverzögerung werden durch den Jitterparameter (ms) spezifiziert. Die Zuverlässigkeitsklasse wird über eine Klasse nach obiger Tabelle ([23]) ausgedrückt. Die Eckdaten des Gruppenmanagements sind über *max_member*, *min_member* und den Zuverlässigkeitsgrad anzugeben.

3. Xpress Transfer Protocol XTP

XTP [37] ist ein leichtgewichtiges, echtzeitfähiges Transferprotokoll und wurde als Teil des *Protocol Engine Projects* [13] bei der Firma *Protocol Engines Incorporated* entwickelt. Eines der wichtigsten Entwurfsziele von XTP war, eine Implementierung des Protokolls in VLSI-Technik zu ermöglichen bzw. zu erleichtern. Durch Zusammenfassung der Transport- und Vermittlungsschicht und Ausnutzung der hohen Geschwindigkeit und Parallelität moderner VLSI Implementierungen soll XTP in der Lage sein, die Datenübertragungsrate moderner Hochgeschwindigkeitsnetze Ende-zu-Ende zu unterstützen. Dieses Ziel soll erreicht werden, ohne einen Kompromiß bezüglich der Zuverlässigkeit und Funktionalität eingehen zu müssen.

Bestehende Protokolle wie TCP/IP oder ISO/TPx wurden nicht für Hochgeschwindigkeits-Netze entwickelt. Annahmen und Einschränkungen über Anwendungen und Netzcharakteristik, wie sie bei dem Entwurf dieser Protokolle gemacht wurden, treffen hier nicht mehr allgemein zu. Obwohl sie viele notwendige Funktionen wie Fehlererkennung, Flußregelung und Reihenfolgesicherung bereits enthalten, fehlen wichtige Mechanismen. Beispielsweise unterstützen sie keine *rate-control* oder selektive Übertragungswiederholung. Ein zuverlässiger *Multicast*-Dienst wird nicht angeboten. Das TPDU-Format ist komplex und erfordert umständliches *parsing* durch variable *Header*-Längen und Mehrfachbelegungen von Feldern. Das Erreichen eines hohen Parallelitätsgrades einer Implementierung ist aufgrund der verwendeten Protokollmechanismen nur bedingt möglich.

Neben der Möglichkeit, XTP in Hardware zu implementieren, werden in XTP Protokollmechanismen verwendet, die eine zuverlässige, realzeitähnliche Datenübertragung auch in Hochgeschwindigkeitsnetzen unterstützen. Unter realzeitähnlicher Verarbeitung wird nach der XTP Protokoll Definition die Fähigkeit verstanden, die Bearbeitungszeit für TPDUs in den Protokollinstanzen auf Sender- und Empfängerseite auf einen Zeitwert kleiner der TPDU-Übertragungsdauer zu begrenzen (bzw. NPDU-Übertragungsdauer bei Verwendung von XTP als Transferprotokoll - siehe [37], Seite 6).

Daneben bietet XTP einen *Multicast*-Dienst, sowie Fehler-, Fluß- und *Rate*-Kontrollmechanismen - ähnlich denen in anderen modernen Transportprotokollen[1]

Um die geforderte, realzeitähnliche Verarbeitung von PDUs zu ermöglichen wurde einerseits das PDU-Format optimiert, andererseits die Idee [12] aufgegriffen, die Funktionalität der Schichten 3 und 4 des ISO-OSI-Referenzmodells in einer Schicht, dem sog. *transfer layer* zusammenzufassen.

Durch diese Integration kann die Protokollbearbeitungszeit verkürzt werden, da in Transport- und Vermittlungsschicht redundant auftretende Funktionen wie beispielsweise Flußre-

---

[1] siehe beispielsweise TP++, VMTP, NETBLT.

gelung und Fehlerbehandlung vermieden werden können. Diese Zusammenfassung von Schicht 3 und Schicht 4 dürfte im Umfeld von Weitverkehrsnetzen jedoch nur schwer zu realisieren sein, da hier die Funktionen der Vermittlungsschicht im Allgemeinen vom Netzbetreiber zur Verfügung gestellt werden. Aufgrund dieser Tatsache werden im folgenden primär Protokollfunktionen mit End-zu-Endsignifikanz (Transportprotokollfunktionen) untersucht.

4. TPDU-Format

Eine XTP TPDU besteht aus einem XTP *Header*, einem mittleren Segment und einem XTP *Trailer*. Das mittlere Segment besteht entweder aus einem Informationssegment oder aus einem Kontrollsegment. Die Syntax für den XTP*Header* und *Trailer* ist grundsätzlich für alle TPDUs gleich: 40 Byte *Header* und 4 Byte *Trailer*. Die Längen der Informations- bzw. Kontrollsegmente sind variabel. Die Länge jedes dieser Segmente - *Header, Trailer* und mittleres Segment - ist immer ein Vielfaches von 4 Bytes (*4 byte alignment*). Hierdurch ist eine besonders effiziente Implementierung auf Systemen mit 32-Bit Wortbreite möglich.

Der XTP Header enthält Informationen zur Steuerung folgender Vorgänge und Protokollfunktionen:

- Identifizierung von TPDUs (*key*)
- Vermittlung von TPDUs durch ein Internet (*route*)
- Begrenzung der Lebensdauer von TPDUs (*ttl*)
- Verwaltung der Reihenfolgenummern (*seq, dseq*)
- Durchführung von *Scheduling* (*sort*)
- Segmentierung (*dlen*)
- Fehlerbehandlung (*hcheck, sync*)

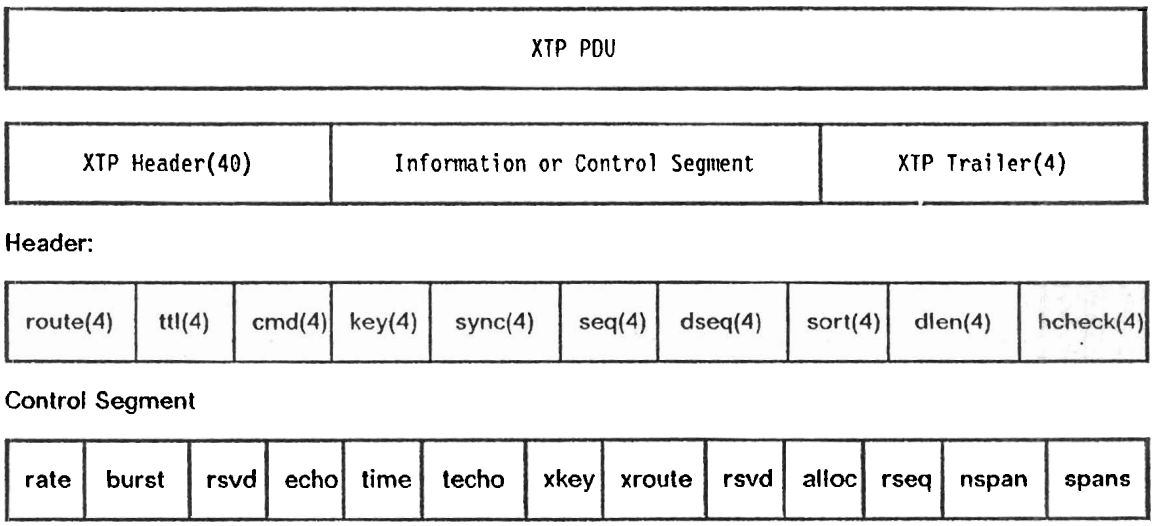

Abbildung 1. XTP-PDU Format

TPDUs, die ein Kontrollsegment beinhalten (sog. Kontroll-TPDUs) dienen dem Austausch von Zustandsinformation zwischen XTP-Protokollinstanzen.

Informationssegmente können sowohl Benutzerdaten eines Transportdienstbenutzers wie auch Protokolldaten (z.B. Adressegment oder Informationen zur Fehlerdiagnose) enthalten. In Informationssegmenten wird neben dem herkömmlichen Datenfeld zusätzlich ein *btag*-Feld (8 Byte) zur Verfügung gestellt, in dem Kontrollinformationen des Dienstbenutzers getrennt von den sonstigen Benutzerdaten übertragen werden können. Die Informationenen im *btag*-Feld werden von XTP wie die übrigen Benutzerdaten transparent übertragen.

Bei den hier betrachteten multimedialen Anwendungen könnte dieses *btag*-Feld vom Benutzer beispielsweise zur Synchronisation mehrerer voneinander abhängiger Kommunikationsbeziehungen verwendet werden. Ebenfalls wäre denkbar, daß der Dienstbenutzer das *btag*-Feld zur Übertragung von Zeitstempeln benutzt. Anhand solcher Zeitstempel könnte der Transportdienstbenutzer bei zeitkritischen Daten feststellen, ob die maximal zulässige Übertragungsdauer einer PDU überschritten wurde.

Aufgrund des Anforderungsprofils in Kapitel 2 ist jedoch für die Überwachung bzw. Einhaltung der Grenzwerte für Übertragungsverzögerung und Jitter das Transportsystem zuständig und nicht der Transportdienstbenutzer: Garantierte Grenzwerte für die End-zu-End-Übertragungsverzögerung können nur in Schicht 3 unterstützt werden, da nur hier die benötigten Informationen über die an der Übertragung beteiligten Teilnetze zur Verfügung stehen. So hat beispielsweise die Wegewahl (*routing*) großen Einfluß auf die Übertragungsverzögerung. Die Sicherstellung der geforderten Bandbreite kann ebenfalls nur in Schicht 3 erfolgen, da es nur hier möglich ist, die entsprechenden Netzressourcen in allen beteiligten Teilnetzen zu reservieren.

Neben der absoluten End-zu-End-Verzögerung spielt bei der Unterstützung multimedialer Anwendungen die Varianz der Verzögerung - auch als Jitter bezeichnet - eine wichtige Rolle. Transportsysteme sollten in der Lage sein, den Jitter durch geeignete Kompensationsmechanismen zu reduzieren. Jitter-Kompensation kann durch Zwischenspeichern der empfangenen Benutzerdaten erreicht werden. Hierzu wird zum einen ein Zwischenspeicher mit ausreichender Größe und zum anderen die Kenntnis über die aktuelle Verzögerung übertragener PDUs benötigt. Die Ermittlung der aktuellen End-zu-End-Übertragungsverzögerung kann grundsätzlich nur mit Hilfe von Zeitstempeln in den übertragenen TPDUs erfolgen.

Soll die Jitter-Kompensation in Schicht 4 erfolgen, müßte das XTP Protokoll um diese Funktion erweitert werden [1]: Das TPDU Format könnte um ein entsprechendes Feld ergänzt oder ein bereits vorhandenes Feld zweckentfremdet werden. Zur Zweckentfremdung bietet sich das bereites erwähnte *btag*-Feld an, das dem Dienstbenutzer dann nicht mehr zur Verfügung stehen würde oder alternativ das *sort*-Feld, das bei nicht gesetztem SORT-Bit bisher unbenutzt ist. Zusätzlich müßte XTP um eine Pufferverwaltung mit entsprechendem Timer-Mechanismus erweitert werden. Grundsätzlich könnte die Jitter-Kompensation auch in Schicht 3 erfolgen - jedoch sprechen folgende Argumente für eine Kompensation in Schicht 4:

Der für Jitter-Kompensation benötigte Speicherbedarf in Endsystemen wächst proportional mit der Anzahl der Transportverbindungen die eine Jitter-Kompensation benötigen. Wenn eine Kompensation nach jeder Teilstrecke erfolgt, wird zwar der benötigte Zwischenspeicher pro Netzverbindung in Schicht 3 kleiner sein (die zu glättenden Zeiten sind geringer), diese Einsparrung kann in einem Vermittlungsknoten (*Gateway*) jedoch schnell zunichte gemacht werden, wenn mehrere (z.B. hunderte) Netzverbindungen eine Jitter-Kompensation benötigen.

5. Verbindungsmanagement

Beginnend mit dem Verbindungsmanagement werden in den folgenden sechs Kapiteln die wichtigsten Aspekte eines Kommunikationsprotokolls am Beispiel XTP für Multimedia im Detail diskutiert.

Verbindungen werden in XTP grundsätzlich implizit [35] mit der Übertragung der ersten TPDU (sog. FIRST-TPDU) zwischen den beteiligten Transportprotokollinstanzen aufgebaut. Die FIRST-TPDU enthält ein Adreßsegment und optional zusätzlich ein Datensegment. Das Adreßsegment beinhaltet Adressierungsinformationen der Sender- und Empfängerinstanz. Es werden unterschiedliche Adressierungsformate, z.B. Internet -, ISO-, XNS-, IEEE 802 Source Route Address -, IP Source Routes Address - und XTP Direct (*locally-defined*) Address - Format unterstützt. Aufgabe des Adreßsegments ist neben der Adressierung von Sender- und Empfängerinstanz die Festlegung des Diensttyps, die Etablierung eines Pfades für Daten- und Kontroll-TPDUs sowie die Festlegung der vom Sender gewünschten Dienstgütewerte für *rate*, *burst* und *maxdata* (siehe Flußkontrolle und Flußsteuerung sowie Dienstgüte).

XTP erlaubt dem Dienstbenutzer, schon während der Verbindungsaufbauphase Daten zu übertragen. Das bedeutet, daß der Dienstbenutzer mit der Übertragung sofort beginnen kann, ohne zuvor auf eine Bestätigung der Verbindungsaufbauanforderung warten zu müssen. Hierdurch können zusätzliche Wartezeiten beim Verbindungsaufbau vermieden werden.

Nach dem Verbindungsaufbau erfolgt die Zuordnung einzelner TPDUs zu einer Verbindung mit Hilfe des *key*-Wertes im Header-Segment. Hierduch kann zum einen Bandbreite (Adress-Segment nur in der FIRST-TPDU enthalten) eingespart und zum anderen die erforderliche Verarbeitungszeit (*context lookup*) verkürzt werden.

Um die Bearbeitungszeit weiter zu verkürzen, sieht XTP beim Sender die Segmentierung von TSDUs in mehrere TPDUs vor. Der Sender kennt nach erfolgreicher Übertragung der FIRST-TPDU die maximale Größe (*maxdata*) einer TPDU die - ohne zusätzliche Segmentierung - durch alle, auf dem gesamten Pfad zur Empfängerseite beteiligten Netze unterstützt wird. Hierdurch wird eine sonst ggf. erforderliche mehrmalige Segmentierung an Netzwerkgrenzen vermieden.

Im Dienst-Feld kann vom Benutzer der geforderte Diensttyp festgelegt werden. Mögliche Diensttypen sind:

- Verbindungsorientierter Dienst
- Transaktion
- unbestätigtes Datagramm
- bestätigtes Datagramm
- isochroner Strom
- Massendaten

Durch die Festlegung des Diensttyps werden die an der Übertragung beteiligten Systemkomponenten über das zu erwartende Kommunikationsmuster informiert. Ensprechend können die benötigten Kommunikationsressourcen reserviert werden. Beispielsweise muß beim isochronen Strom garantiert werden, daß die angeforderte Bandbreite (ensprechend der *rate* und *burst* Werte im Adreß-Segment) als konstanter Wert anzusehen ist und nicht - wie

z.B. beim normalen verbindungsorientierten Dienst möglich - im Überlastfall kurzzeitig reduziert werden kann.

Kann die Partnerinstanz die gewünschten Anforderungen nicht garantieren, wird dies dem Sender der FIRST-TPDU durch eine entsprechende Diagnose-TPDU (DIAG TPDU) angezeigt und der Verbindungsaufbau wird abgelehnt.

Die Festlegung des Diensttyps beim Verbindungsaufbau mit den entsprechenden Dienstgüteparametern (*rate-*, *burst-* und *maxdata*-Werte) eignet sich grundsätzlich auch für **multimediale** Anwendungen. Um eine praxisgerechte Aushandlung der Dienstgüte zu ermöglichen, sollte jedoch neben der Angabe der im ungünstigsten Fall noch akzeptierbaren Werte auch die Angabe der vom Dienstbenutzer gewünschten Dienstgütewerte möglich sein (z.B. *des_rate_req, des_burst_req*). Entspechend dem Anforderungsprofil in Kapitel 2 sollten zur Unterstützung multimedialer Anwendungen noch weitere Parameter eingeführt werden (z.B. *end to end delay* und *jitter*), um eine präzisere Spezifikation der geforderten Dienstgüte zuzulassen.

6. Prioritätsverfahren

XTP unterstützt ein prioritätsgesteuertes Planungsverfahren (*priority scheduling*) um die Bearbeitungsreihenfolge empfangener bzw. zu sendender TPDUs unterschiedlicher Verbindungen kontrollieren zu können. Zu diesem Zweck sieht XTP ein *sort*-Feld (32-Bit) im *Header* vor, das nur dann interpretiert wird, wenn das SORT-Bit (*cmd*-Feld im *Header*) gesetzt ist. Durch das *sort*-Feld werden 2^{32} unterschiedliche Prioritätsklassen festgelegt, wobei *sort* = 0 die höchste und *sort* = 2^{32} - 1 die niedrigste Klasse darstellt. TPDUs mit höherer Priorität werden vor TPDUs mit niedrigerer Priorität bearbeitet. TPDUs, in denen das SORT-Bit nicht gesetzt ist, werden erst nach TPDUs mit gesetztem SORT-Bit bearbeitet. Die Interpretation des *sort*-Feldes ist optional, XTP-Implementierungen müssen das Feld nicht zwingend bei der Bearbeitung berücksichtigen.

Bei der Verarbeitung der TPDUs sollte grundsätzlich nach 2 Klassen unterschieden werden:

1. **Daten mit Echtzeitanforderungen:** Bearbeitung gemäß eines Echtzeitplanungsverfahrens wie beispielsweise *earliest deadline first* oder *rate monotonic*.
2. **Daten ohne Echtzeitanforderungen:** Die nach der Verarbeitung übrigbleibende Verarbeitungskapazität soll für die Bearbeitung dieser Daten verwendet werden. Hier kann ein prioritätsgesteuertes Verfahren über das *sort*-Feld verwendet werden.

Die Wahl der Prioritätswerte erfolgt in XTP lokal bei der Senderinstanz. Die Abbildung der Dienstanforderungen auf die Prioritätswerte kann jedoch nur bei Kenntnis der Netzauslastung in allen an der Kommunikation beteiligten Teilnetze erfolgen und sollte daher in der Vermittlungsschicht erfolgen. In XTP sind die dazu erforderlichen Funktionen - auch bei Betrachtung des Gesamtfunktionsumfangs (Schicht 3 und 4) - nicht enthalten. Die Verwendung der Prioritätsmechanismen zur Sicherstellung der Echtzeitanforderungen *multimedialer* Anwendungen ist daher aus unserer Sicht nicht möglich.

7. Fehlerbehandlung

Wenn die Zuverlässigkeitsanforderungen der Transportdienstbenutzer nicht bereits durch den Vermittlungsdienst erfüllt werden, dann müssen in der Transportschicht Mechanismen zur Fehlererkennung, -meldung und -behebung bereitgestellt werden.

Fehlererkennung
Bitfehler werden in XTP mit Hilfe von zwei unterschiedlichen Prüfsummen erkannt. Eine der Summen wird über dem *Header*, die andere über dem mittleren Segment gebildet. Die *Header*-Prüfsumme wird grundsätzlich immer berechnet, die Berechnung der Prüfsumme über dem mittleren Segment kann mit Hilfe des NOCHECK-Bits im *cmd*-Feld des *Headers* kontrolliert werden.

Die Erkennung von **TPDU-Verlusten, Duplikaten oder Reihenfolgevertauschungen** erfolgt mit Hilfe von Reihenfolgenummern. XTP numeriert die übertragenen Datenbytes modulo 2^{32}. Das *seq*-Feld des *Header* enthält die Reihenfolgenummer des ersten Bytes im Daten-Segment. Zusammen mit dem *dlen*-Feld können die bisher empfangenen Benutzerdaten identifiziert und Lücken erkannt werden. Die Überwachung der Reihenfolgenummern kann durch das NOERR-Bit im *cmd*-Feld des Headers gesteuert werden.

Fehlermeldung
Mechanismen für die Fehlermeldung ermöglichen dem Empfänger, den Sender über erkannte Fehler zu unterrichten. XTP bietet hierzu zwei unterschiedliche Verfahren an: *go-back-n* und *selective reject*. Beim *go-back-n* Verfahren unterrichtet der Empfänger den Sender über die Reihenfolgenummer, von der an Daten verloren gingen oder vermißt werden. Detailliertere Informationen werden beim *selective reject* mitgeteilt. Hier werden alle vermißten Daten dem Sender durch Bereichsangaben (*gaps*) mitgeteilt.

Grundsätzlich fordert in XTP die Senderinstanz die Empfängerinstanz auf, Fehler zu melden. Dies geschieht durch Senden einer TPDU mit gesetztem *sreq*- oder *dreq*-Bit. Neben dieser Methode existiert noch das *fast-negative-acknowledgment* Verfahren (*FASTNAK*-Bit im *cmd*-Feld des *Headers* gesetzt). Hierbei informiert die Empfängerinstanz die Senderinstanz unmittelbar nach Erkennung eines Fehlers.

Fehlerbehebung
XTP verwendet zur Fehlerbehebung ausschließlich die Übertragungswiederholung. Dabei wird die Information über verlorene bzw. verfälscht empfangene Daten verwertet. Wenn der Empfänger nicht in der Lage ist, Daten, die außerhalb der Reihenfolge empfangen wurden, zwischenzuspeichern, kann in XTP das *go-back-n* Verfahren benützt werden. Hierbei werden alle Daten ab einer bestimmten Reihenfolgenummer nochmals übertragen. Obwohl dieses Verfahren die Protokollbearbeitung vereinfachen kann, führt es leicht zur Verschwendung von Bandbreite, insbesondere bei Netzen mit einer hohen Übertragungskapazität und einer großen Übertragungsverzögerung (z.B. Satelitennetzwerke) [20]. Neben diesem Verfahren bietet XTP auch noch die *selektive* Übertragungswiederholung an. Dabei werden nur diejenigen Daten nochmals gesendet, die der Empfänger nicht korrekt oder garnicht empfangen hat. Durch Tupel von Reihenfolgenummern wird dem Sender mitgeteilt, welche Daten korrekt empfangen wurden. Der Sender hat dann die Aufgabe, aufgrund dieser Information, diejenigen Daten zu bestimmen, die nochmals übertragen werden müssen. Dieses Verfahren spart Bandbreite ein, benötigt aber beim Empfänger große Zwischenspeicher und erhöht insgesamt die Komplexität des Protokolls.

Durch die beschriebenen Mechanismen eignet sich XTP prinzipiell auch zur Unterstützung multimedialer Anwendungen. Aufgrund seiner hohen Flexibilität können alle in Kapitel 2 aufgeführten Zuverlässigkeitsklassen unterstützt werden.

Da im allgemeinen die zu übertragenden Daten multimedialer Anwendungen zeitkritische Datenströme darstellen, kann jedoch unter Umständen das Verfahren der Fehlerbehebung durch Übertragungswiederholung nicht angewendet werden. Die maximale, vom Dienstbenutzer akzeptierte End-zu-End-Übertragungsverzögerung kann gegebenenfalls eine erneute Datenübertragung im Fehlerfall unmöglich machen. Zur Vermeidung dieses Problems bietet sich eine *forward error correction* Methode an. Hier können in Abhängigkeit der zu erwartenden Fehlercharakreristik (Bitfehler-, Bitbündelfehlerrate, TPDU Verlustrate) unterschiedliche Verfahren zum Einsatz kommen (z.B. das der *Cross Interleaved Reed Solomon Code* aus der CD-DA-Technologie oder andere fehlerkorrigierende Codes [6]). Alle diese Mechanismen verbrauchen jedoch zusätzliche Bandbreite und Rechenzeit.

Zusätzlich zu den genannten Fehlerarten sollten im Umfeld multimedialer Anwendungen auch Fehler durch verspätet eintreffende TSDUs erkannt und ggf. dem Benutzer angezeigt werden. Die hierzu erforderlichen Protokollerweiterungen wurden bereits im Abschnitt „TPDU-Format" beschrieben.

8. Flußregelung und Flußsteuerung

Flußregelung bzw. Flußsteuerung ist ein wichtiger Gesichtpunkt in Hochgeschwindigkeitsnetzen, da ein erhöhtes Risiko besteht, daß ein Empfänger bzw. ein Vermittlungsrechner durch die hohe Übertragungsgeschwindigkeit mit Daten überflutet wird. In diesem Zusammenhang wurde beobachtet, daß der Hauptgrund für Datenverluste in Hochgeschwindigkeitsnetzen eben in einer solchen Überlastung von Vermittlungsrechnern zu sehen ist.

XTP bietet zwei verschiedene, sich ergänzende Mechanismen zur Flußsteuerung. Dies ist zum einen das klassische Schiebefensterverfahren, in dem der Datenempfänger, ausgehend von seinem vorhandenen Empfangspufferplatz Senderechte explizit durch Übermittlung von Sequenznummern an den Sender vergibt.

Der zweite Mechanismus, in XTP als *rate control* bezeichnet, betrachtet nicht den Pufferplatz, sondern die Verarbeitungsleistung des Empfängers [byte/s] als kritische Größe (bei Einsatz als *Transfer*-Protokoll wird der im Netz realisierbare Durchsatz als weitere Größe hinzugenommen). Die protokollinterne Realisierung erfolgt über drei Variable *credit*, *burst* und *refresh-timer*. Ausgehend von einem zyklischen Verhalten, wie es im Multimediakontext meist anzutreffen ist, bezeichnet *refresh-timer* die Länge eines Zyklus [s], *burst* die maximale Menge der in einem Zyklus zu übertragenden Daten [byte] an. Die lokale Variable *credit* gibt zu jedem Zeitpunkt die bis zum nächsten Ablauf des *refresh-timer* noch übertragbare Datenmenge in Bytes an.

Die beiden Parameter *burst* und *refresh-timer* werden beim Verbindungsaufbau zwischen Sender und Empfänger ausgehandelt. Während der laufenden Übertragung wird die Einhaltung der Absprachen jedoch allein durch den Sender ohne Mitwirkung des Empfängers erzwungen[2].

---

[2] Erläuterung: Mit jeder übertragenen TPDU wird der Wert von *credit* um die Anzahl der in der TPDU enthaltenen Benutzerdaten verringert. Die Datenübertragung hält an, wenn der Wert von *credit* kleiner oder gleich Null wird. Nach jeder Periode des *refresh-timer* wird der Wert von *credit* wieder auf den ursprünglichen Wert von *burst* gesetzt. Ein Wert von Null für *burst* bedeutet, daß keine Flußsteuerung stattfindet, wohingegen ein Wert von Null für *rate* die Datenübertragung anhält. Kontroll-TPDUs dürfen jedoch auch in diesem Fall weiter

Wird XTP als *Transfer Layer Protocol* zur Erbringung von Vermittlungs- und Transportfunktionen eingesetzt, dürfen Vermittlungsrechner die Werte für *rate* und *burst* in Kontroll-TPDUs verändern. Zudem brauchen Vermittlungsrechner nicht auf eine passierende Kontroll-PDU zu warten, sondern können zu jeder Zeit eine sog. *Route-Control*-PDU (RCNTL-PDU) mit entsprechenden Werten für *rate* und *burst* ausgeben. Wenn ein Vermittlungsrechner *rate*- und *burst*-Werte in passierenden PDUs für sämtliche Verbindungen registriert, so ist er in der Lage, die Summe dieser Werte zu bilden. Diese Summe bestimmt den Bandbreitenbedarf des Vermittlungsrechnes. Durch Modifikation der Kontroll-PDUs und der Fähigkeit der Summenkontrolle, ist der Vermittlungsrechner in der Lage, Bandbreite zu verwalten.

Da die erforderliche Bandbreite und das zeitliche Verhalten der einzelnen zu übertragenden TSDUs in multimedialen Anwendungen vorgegeben sind, ist eine Flußregelung mittels Schiebefenster zwischen den Partnerinstanzen nicht sinnvoll anwendbar, sondern nur eine Flußsteuerung via *rate control*. In dieser Beziehung eignet sich XTP, da die Flußregelung ausgeschaltet werden kann („NOFLOW" Steuerfeld gesetzt) und eine Flußsteuerung protokollintern vorhanden ist. Für multimediale Anwendungen sollte jedoch auch eine Flußsteuerung am Dienstzugangspunkt vorgesehen werden, da ein „Wohlverhalten" des Dienstbenutzers am TSAP im allgemeinen nicht angenommen werden kann. Unter „Wohlverhalten" wird hier verstanden, daß ein Dienstbenutzer periodisch auftretende Daten dem Transportsystem stets auch absprachegemäß mit derselben Periode übergibt.

9. Gruppenkommunikation

XTP definiert einen *Multicast*-Modus, in dem ein Sender TPDUs gleichzeitig zu einer Gruppe von Empfängern überträgt (unidirektionales *one-to-many*).

Der *Multicast*modus gleicht in vieler Hinsicht dem Modus für einen einzelnen Empfänger. Der Sender gibt eine FIRST-TPDU und nachfolgend Daten-TPDUs aus. Eine Fehlerbehandlung ist vorgesehen. Für die Übertragungswiederholung wird das Verfahren *go-back-n* eingesetzt. Selektive Übertragungswiederholung wird nicht unterstützt. Da in der Menge der Empfänger die Größe der verfügbaren Empfangspuffer sehr unterschiedlich sein kann, wird die Datenübertragung mit der Verarbeitungsgeschwindigkeit des langsamsten Empfängers durchgeführt. Entdeckt ein Empfänger eine TPDU außerhalb der Reihenfolge, so wird eine Kontroll-TPDU (sog. *reject packet*) ausgegeben. Diese Kontroll-TPDU wird an die Menge aller an der Verbindung beteiligten Instanzen inklusive aller Empfänger übertragen, um sie über das Auftreten eines Fehlers zu informieren. Würden alle Fehler grundsätzlich von jedem Empfänger gemeldet bei dem sie erkannte wurden, ließe sich die Überschwemmung des Senders mit ankommenden Kontroll-TPDUs (*reject packets*) kaum vermeiden. Um diesen Effekt einzudämmen, dürfen die Empfänger keine Kontroll-TPDU senden, wenn sie davon ausgehen können, daß der Sender bereits über den Fehler informiert wurde. Während ein Empfänger eine Kontroll-TPDU erstellt und auf die Übertragung wartet, horcht er das Netz nach anderen Kontroll-TPDUs ab. Kommt eine andere Kontroll-TPDU für denselben Sender an, vergleicht der Empfänger seinen Wert für die Übertragungswiederholung mit dem Wert in der gerade empfangenen TPDU. Wird sein Wert durch den der TPDU abgedeckt, kann er seine eigene Kontroll-TPDU verwerfen (sog. *damping* Verfahren). Andernfalls schickt er seine Kontroll-TPDU zum Sender.

---

ausgegeben werden. Die Werte für *rate* und *burst* gelten jeweils für eine Richtung der Datenübertragung. So sind in Abhängigkeit der Übertragungsrichtung unterschiedliche Übertragungsraten möglich.

Neben diesem Verfahren bietet XTP auch eine unzuverlässige Datenübertragung im Multicast Modus. Bei diesem *no error* Modus verwirft ein Empfänger jene TPDUs die verfälscht sind. Er informiert lediglich den Dienstbenutzer über das Ereignis.

Aufgrund der oben beschriebenen Verfahren bietet XTP folgende Zuverlässigkeitsgrade für die Gruppenkommunikation [17]:

Zuverlässigkeitsgrad k = 0, die *Multicast*-Datenübertragung wird als erfolgreich angesehen, auch wenn kein Empfänger die TPDUs korrekt erhalten hat.

Zuverlässigkeitsgrad k = 1, die *Multicast*-Datenübertragung wird als erfolgreich angesehen, wenn mindestens 1 Empfänger die TPDUs erhalten hat

Die Fehlerkorrektur erfolgt durch wiederholtes Senden der verlorenen TPDUs an die gesamte *Multicast*-Gruppe.

Da in XTP keine Mechanismen für die Festlegung und Verwaltung von *Multicast*-Gruppen vorgesehen sind, ist eine Erweiterung des Zuverlässigkeitsgrads auf k = 2 bis k = Gruppengröße nicht ohne Änderung des Protokolls möglich [25]. XTP beinhaltet zwar einen Mechanismus, der es einem Empfänger ermöglicht, an einer bestehenden *Multicast*-Kommunikationsbeziehung teilzunehmen (*join group* Funktion). Das Eintreten eines weiteren Empfängers in die Gruppe wird jedoch nicht verwaltet. Zudem müssen beim Sender ankommende Kontroll-TPDUs eindeutig dem jeweiligen Initiator zugeordnet werden können (z.B. zur Feststellung ob ein Empfänger ausgefallen ist). Das bedeutet, daß das *damping* Verfahren nicht angewandt werden kann und, daß die Kontoll-TPDUs eine Identifikation des jeweiligen Initiators enthalten müssen. Zudem muß jede dynamische Veränderung der Gruppe dem Sender mitgeteilt werden.

Der von XTP erbrachte *Multicast*-Dienst eignet sich für reines *Broadcast* (z.B. Nachrichtenverteildienst, Zuverlässigkeitsgrad k = 0) und zur Suche einer bestimmten Information in einem verteilten System (z.B. finde einen freien Rechner, Zuverlässigkeitsgrad k = 1).

Ein großer Nachteil ist jedoch die nur unidirektionale *one-to-many* Kommunikationsbeziehung. XTP sollte neben den oben aufgeführten Erweiterungen auch bezüglich eines bidirektionalen *one-to-many Multicast* modifiziert werden.

10. Dienstgüteparameter

XTP unterstützt verschiedene Dienstgüteparameter, die nicht immer für die Multimedia-Kommunikation ausreichen. Im Folgenden seien die wichtigsten kurz aufgeführt:

Dienstgüte-Parameter	*Anmerkungen*
Verzögerung	In XTP werden nur Durchschnittswerte für die End-zu-End-Verzögerung festgelegt. Es sind keine Garantien bzw. Mechanismen vorgesehen um die angegebenen Wert einzuhalten. Dies ist somit nicht für die Kommunikation von Audio- und Videodaten geeignet. Man beachte hier insbesondere die Anforderungen von Seiten eines Dialogdienstes (wie in Abschnitt 2 beschrieben).

Zuverlässigkeitsklasse	Der Dienstbenutzer gibt beim Verbindungsaufbau eine Zuverlässigkeitsklasse an. Die entsprechende Fehlerbehandlung wird durchgeführt. (Siehe Abschnitt zur Fehlerbehandlung, in dem die Implikation bezüglich Multimedia beschrieben wurden).
TPDU Größe	XTP kennt nach Übertragung der FIRST-PDU die maximale TPDU-Größe, die ohne zusätzliche Fragmentierung an Subnetzgrenzen übertragen werden kann. Hier sehen wir bisher keine Problematik im Kontext von Audio- und Videokommunikation.
impl. Verbindungsaufbau	XTP ermöglicht dem Dienstbenutzer bereits während des Verbindungsaufbaus Benutzerdaten zu übertragen. Diese Daten sind nur bedingt für ein kontinuierliches Medium einzusetzen, weil die Dauer der Übertragungsverzögerung zwischen der ersten PDU und den folgenden stark variieren kann.
Übertragungsrate	Der Dienstbenutzer gibt seine gewünschte Sendedatenrate (*rate*- und *burst*-Werte) an, XTP führt entsprechend Ratenkontrolle durch. (Anmerkungen hierzu siehe Abschnitt zur Flußkontrolle und Flußsteuerung.
Empfangspuffer	Der Dienstbenutzer gibt Empfangspuffergröße an. XTP führt entsprechend Flußkontrolle auf Anwendungsebene durch (*Reservation*-Modus). Dies muß mit dem Buffermanagement des Multimedia-Systems geeignet zusammenarbeiten.

11. Ausblick

XTP eignet sich nach dem in diesem Bericht aufgezeigten Punkten in seiner jetzigen Form nur bedingt zur Kommunikation kontinuierlicher Daten. Als Beispiele für Problembereiche seien hier noch einmal die Überwachung der End-zu-End Verzögerung und des Jitters erwähnt. Als wünschenswerte, und ab einer gewissen Gruppengröße notwendige, Erweiterung sehen wir im Hinblick auf *Multicast* auch eine eigene Gruppenverwaltung oder eine geeignete Schnittstelle zu externen *Directory*-Diensten an.

Mit den in diesem Papier aufgezeigten Erweiterungen ist jedoch im wesentlichen eine multimediale Kommunikation möglich. Wegen des konkreten Mangels an einem globalen Reservierungsmechanismus für netzinterne Betriebsmittel, wie er beispielsweise bei ST II geboten wird, haben wir uns entschieden weiter ST II in der Vermittlungsschicht als grundlegendes Protokoll zu verwenden. An der Definition von XTP als reinem Transportprotokoll, das dann oberhalb ST II eingesetzt wird, wird zur Zeit am ENC gearbeitet.

Literaturverzeichnis

[1] *G. Anastasi, M. Conti, E.Gregori;* **TPR: A Transport Protocol for Real Time Services in an FDDI Environment;** In: M. Johnson (Hrsg.); Protocols for High-Speed Networks; Elsevier Science Publishers B.V. (North Holland), IFIP, 1990

[2] *S. R. Ahuja, J. Robert Ensor, David N. Horn;* **The Rapport Multimedia Conferencing System;** Proceedings of the Conference on Office Information Systems, Palo Alto CA, March 1988, pp. 1-8.

[3] *David P. Anderson, Ralf Guido Herrtwich;* **Internet Communication with End-to-End Performance Guarantees;** Informatik-Fachberichte, no. 293, Springer Verlag, 1991.
[4] *Heinrich Armbrüster, Hans Jörg Rothamel;* **Breitbandanwendungen und -dienste: Qualitative und quantitative Anforderungen an künftige Netze;** ntz, vol. 43, no. 3, 1990, pp. 150-159.
[5] *Bell Telephone Labs.;* **Engineering and Operations in the Bell System;** AT&T Bell Labs, 1984.
[6] *Ernst W. Biersack;* **Performance Evaluation of Forward Error Correction in ATM** Computer Communication Review, ACM SIGCOMM, Vol.22, No.1, Januar 1992.
[7] *Steward Brand;* **The Media Lab, Inventing the Future at MIT;** Viking Penguin, 1987.
[8] *Timothy S. Balraj, Yechiam Yemini;* **PROMPT - A Destination Oriented Protocol for High Speed Networks;** M. Johnson (Hrsg.); Protocols for High-Speed Networks; Elsevier Science Publishers B.V. (North Holland), IFIP, 1990
[9] *E.W. Biersack, C.J. Cotton, D.C. Feldmeier, A.J. McAuley;* **An Overview of the TP++ Transport Protocol Project;** submitted for publication
[10] *Ernst W. Biersack;* **Connection Management using Synchronized Clocks;** Proc. of Third IFIP WG 6.4 Conference on High Speed Networking, Berlin, March 1991, pp.225-238
[11] *International Telegraph and Telephone Consultative Committee;* **B-ISDN Service Aspects;** CCITT Study Group XVIII, Draft Recommendation I.211, Geneva, 23-25 May 1990.
[12] *CELAR (Centre d'Electronique de L'Armement);* **Military real time local area network;** Rennes, France, Februar 1987
[13] *Greg Chesson;* **The Protocol Engine Project;** UNIX Review, Vol.5, No.9, September 1987, pp.70-77
[14] *Greg Chesson;* **The Evolution of XTP;** Proc. of Third IFIP WG 6.4 Conference on High Speed Networking, Berlin, March 1991, pp.15-24.
[15] *David R. Cheriton, Erik Nordmark;* **Experiences from VMTP: How to achieve low response time;** H. Rudin, R. Williamson (Hrsg.); Protocols for High-Speed Networks; Elsevier Science Publishers B.V. (North Holland), IFIP, 1989, pp.43-56
[16] *David D. Clark, Mark L. Lambert, Lixia Zhang;* **NETBLT: A High Throuput Transport Protocol;** Proc. SIGCOMM'87, ACM, August 1987, pp.353-359
[17] *J. Crowcroft, K. Paliwoda;* **A Multicast Transport Protocol;** Trans. SIGCOMM, ACM, August 1988, pp.247-256
[18] *S. Deering;* **Host Extensions for IP Multicasting** Network Working Group, RFC 1112, August 1989
[19] *Willibald Doeringer, Doug Dykeman, Matthias Kaiserswerth, Bernd Meister, Harry Rudin, Robin Williamson* **A Survey of Light-Weight Transport Protocols for High-Speed Networks** IEEE Transactions on Communications, Vol.38, No.11, November 1990, pp.2025-2039
[20] *David C. Feldmeier, Ernst W. Biersack;* **Comparison of Error Control Protocols for High Bandwith-Delay Product Networks** In: M. Johnson (Hrsg.); Protocols for High-Speed Networks; Elsevier Science Publishers B.V. (North Holland), IFIP, 1990
[21] *Domenico Ferrari;* **Client Requirements for Real-Time Communication Services;** International Computer Science Institute, Technical Report 90-007, Berkeley, March 1990.
[22] *G. D. Flinchbaugh, P. L. Martinez, D. S. Rouse;* **Network Capabilities in Support of Multimedia Applications;** IEEE Global Telecommunications Conference & Exhibition (GLOBECOM), Conference Record vol. 1, San Diego, Dec. 2-5, 1990, pp. 322-326.
[23] *Lutz Henckel, Heiner Stüttgen;* **Transportdienste in Breitbandnetzen;** E. Effelsberg, H.W. Meuer, G. Müller (Hrsg.); Proceedings zur GI/ITG Fachtagung "Kommunikation in verteilten Diensten" in Mannheim; Springer Verlag; Februar 1991; pp.96-111
[24] *Dietmar Hehmann, Michael Salmony, Heinrich Stüttgen;* **Transport Services for Multimedia Applications;** Proceedings of the IFIP WG 6.1/WG 6.4 Workshop on Protocols for High Speed Networks, North Holland, 1989, 303-321
[25] *Larry Hughes;* **Multicast Response Handling Taxonomy;** Computer Communications, vol. 12 No 1, Feb. 1989, pp. 39-46.
[26] *Information Processing Systems - Open Systems Interconnection* **Transport service definition;** ISO 8072, 1986
[27] *Information Processing Systems - Open Systems Interconnection* **Transport protocol specification;** ISO 8073, 1986
[28] *Mark G.W. Jones, Soren-Aksel Sorensen, Steve R. Wilbur;* **Protocol design for large group multicasting: the message distribution protocol;** Computer Communications, Vol.14, No.5, June 1991, pp.287-297
[29] *Robert M. Sanders, Alfred C. Weaver;* **The Xpress Transfer Protocol (XTP) - A Tutorial** ACM Computer Communications Review, Vol.20, No.5, October 1990, pp.67-88

[30] *Thomas Schütt, Manny Farber;* **The Heidelberg High Speed Transport System: First Performance Results;** In: 3rd International IFIP WG6.1/6.4 Workshop on Protocols for High-Speed Networks, Stockholm, May 13-15, 1992 pp. 35-50.

[31] *Ralf Steinmetz,* **Multimedia Synchronization Techniques: Experiences Based on Different System Structures;** IEEE Multimedia Workshop '92, Monterey, CA, USA, April, 1992.

[32] *Ralf Steinmetz, Thomas Meyer;* **Modelling Distributed Multimedia Applications;** IEEE Int. WS on Advanced Communications and Applications for HS Networks, München, March 1992.

[33] *Claudio Topolcic;* **Experimental Internet Stream Protocol, Version 2 (ST-II);** RFC 1190, Oct. 1990.

[34] *R.W. Watson;* **Timer-Based Mechanisms in Reliable Transport Protocol Connection Management** Computer Networks, No.5, 1981, pp.47-56

[35] *R.W. Watson;* **The Delta-t Transport Protocol: Features and Experience;** H. Rudin, R. Williamson (Hrsg.); Protocols for High-Speed Networks; Elsevier Science Publishers B.V. (North Holland), IFIP, 1989, pp.3-18

[36] *David J. Wright, Michael To;* **Telecommunication Applications of the 1990s and their Transport Requirements;** IEEE Network Magazine, vol.4, no.2, March 1990, pp.34-40.

[37] *Protocol Engines Incorporated;* **XTP Protocol Definition Revision** 3.6; PEI 92-10, Protocol Engines Incorporated, Santa Barbara, CA 93101, Januar 92

[Hein 92] B. Heinrichs: XTP Specification and Parallel Implementation, Proc. International Workshop on Advanced Communications and Applications for High Speed Networks, pp. 77-84, March 16-19, 1992

[Hein 93] B. Heinrichs, R. Karabek, W. Mers: Optimierung von Transfersystemen, Proc. der ITG/GI-Fachtagung Kommunikation in Verteilten Systemen 93, München, März 93

[ISO 92] HSTP: Proposed Working Draft High Speed Transport Protocol, ISO/IEC JTC1/SC6 Telecommunications and Information Exchange between Systems, Januar 1992

[ISO 88] ISO 8473: Protocol for Providing the Connectionless-mode Network Service (Internetwork Protocol), December 1988

[ISODE] ISODE-8.0 Documentation Set Department of Computer Science UCL, Gower Street, London, WC1E 6BT, UK (or ftp "isode-8-doc.tar.Z" from uu.psi.com [136.161.128.3]).

[Jain 90] N. Jain, M. Schwartz, T.R. Bashkow: Transport Protocol Processing at GBPS Rates, SIGCOMM 90 Symposium Communications Architectures & Protocols, pp. 188-199, September 24-27, 1990

[Jak 91] K. Jakobs: Beyond the Interface - Group Communication Services Supporting CSCW, Proc. IFIP Work. Conf. on Support Functionality in the Office Environment, MEOW-91

[Jak 92] K. Jakobs: Matching X.400 Services Against User Requirements, Proc. Silicon Valley Networking Conference, SVNC-92

[RACE92] RACE II Programme Description, CEC

[Wood 89] C.M. Woodside, J.R. Montealegre: The Effect of Buffering Strategies on Protocol Execution Performance, IEEE Transactions on Communications, vol. 37, no. 6, pp. 545-554, June 1989

[XTP 92] XTP Protocol Definition, Revision 3.6, Protocol Engine Incorporated, Januar 1992

[Zhan 90] X. Zhang, A.P. Seneviratne: An Efficient Implementation of a High-Speed Protocol without Data Copying, Proc. 15th Conference on Local Computer Networks, pp. 443-450, Sep.30 - Oct. 3, 1990

XTP und Multimedia ?

Markus Steffens

Johann Wolgang Goethe-University of Frankfurt
Fachbereich Informatik, Institut für Telematik
Robert-Mayer-Strasse 11-15,
6000 Frankfurt

Jochen Sandvoss, Thomas Schütt, Ralf Steinmetz

IBM European Networking Center,
Tiergartenstr. 8, Postfach 10 30 68,
6900 Heidelberg

Zusammenfassung

Dieser Beitrag untersucht die Eignung der in XTP (Xpress Transfer Protocol) eingesetzten Protokollfunktionalitäten zur Kommunikation von Audio- und Videodaten. Ausgehend von den Anforderungen multimedialer Anwendungen werden die vorhandenen XTP-Mechanismen kurz erläutert. Anschließend werden für die wichtigsten Teilaspekte des Kommunikationsprotokolls (Verbindungsmanagement, Prioritätssteuerung,Flußsteuerung etc.) jeweils im Einzelnen die Anforderung von Seiten der Übertragung kontinuierlicher Medien und mögliche Lösungen im Kontext von XTP diskutiert. Es zeigt sich, daß zur Unterstützung multimedialer Kommunikationsformen sowohl geeignete Interpretationen vorhandener XTP-Protokollmechanismen wie auch Protokollerweiterungen erforderlich sind.

Abstract

This paper analyzes the requirements of multimedia communications in terms of XTP (Xpress Transfer Protocol). In a first step, multimedia communication requirements are described. Subsequently individual features of XTP relevant for audio and video data transfer are presented. We describe the multimedia requirements and provide a possible way to satisfy them using XTP. It turns out that a multimedia capable XTP requires new interpretations of existing mechanisms as well as some extensions.

1. Einleitung

Integrierte Multimedia-Systeme bieten Möglichkeiten zur rechnergestützten Erzeugung, Verarbeitung, Darstellung, Speicherung und Kommunikation unabhängiger diskreter Medien

wie Text oder Graphik sowie kontinuierlicher Medien wie Ton oder Bewegtbild [23]. Solche lokalen Systeme sind heute als Produkte auf dem Markt. Verteilte Multimedia Systeme sind die Basis für eine Menge interessanter Anwendungen. Neben Multimedia-Dialogsystemen entstehen hier auch Verteildienste, Multimedia-Konferenzsysteme [2], oder beispielsweise eine individuelle Zeitung [7]. Diese Anwendungen erfordern als wesentliches Merkmal ein Kommunikationssystem, daß den Anforderungen für Multimedia-Daten genügt.

Das ISO Transportprotokoll kann zur Unterstützung eines solchen multimediafähigen Systems erweitert werden [30]. Auch TCP kann mit einigen Veränderungen kontinuierliche Medien übertragen [3]. Jedoch sind diese Protokolle nicht per se den Eigenschaften von Audio und Video angepaßt, sie wurden nachträglich erweitert. Weitere interessante Entwicklungen sind NetBLT (Network Block Transfer Protocol) vom MIT [16], VMTP (Versatile Message Transfer Protocol) aus der Stanford University [15], TP++ von Bellcore [20] und XTP (Xpress Transfer Protocol) [14; 29; 37]. XTP bietet dabei die Möglichkeit einer Zusammenfassung der Vermittlungs- und Transportschicht.

Im HeiTS-Projekt am Europäischen Zentrum für Netzwerkforschung der IBM verwenden wir zur Zeit zwei alternative Vermittlungsprotokolle: ein modifiziertes X.25 Protokoll sowie das im amerikanischen DoD Kontext definierte *Internet Stream Protocol* ST-II. Die Implementierung erfolgte auf OS/2 bzw. AIX Plattformen. In der nächsten Stufe wird ST-II [33] in das OS/2-System integriert. Aus den mit dem erweiterten X.25 gewonnenen Erfahrungen sollen dabei ggf. Vorschläge für Erweiterungen des ST II Protokolles abgeleitet werden. In diesem Kontext wurde XTP als ein mögliches Transportprotokoll untersucht. In diesem Bericht werden die ersten Erfahrungen hierüber zusammengefaßt.

Ausgehend von den Anforderungen multimedialer Anwendungen wird in den Kapiteln 3 und 4 ein kurzer Überblick zu XTP gegeben. Die folgenden Kapitel 5 bis 10 betrachten einzelne Aspekte des Kommunikationsgeschehens, indem jeweils zuerst der XTP-Mechanismus erläutert wird. Anschließend werden jeweils die Anforderungen an einen solchen Mechanismus aus Sicht der Übertragung kontinuierlicher Medien und eine mögliche Lösung im Kontext von XTP aufgezeigt.

2. Anforderungen multimedialer Anwendungen

Im folgenden Abschnitt werden anhand von zwei Beispielanwendungen typische Anforderungen abgeleitet, die multimediale Anwendungen an ein Kommunikationssystem stellen. Hierbei wird besonders auf die benötigte Funktionalität bezüglich der Transportschicht eingegangen (siehe auch [4; 21; 24; 36]).

Das **ersten Beispiel** einer Anwendung besteht aus **Dialogdiensten** nach [11]. Eine Punkt-zu-Punkt Kommunikationsverbindung überträgt Daten, Graphiken, Sprache und Bewegtbilder zwischen den beiden Anwendungsinstanzen. Zusammenfassend lassen sich folgende Anforderungen ableiten:

1. **Daten:** Bei der Übertragung der Daten muß die Fehlerfreiheit auf Bitebene gewährleistet sein. Die benötigte Bandbreite und Ende-zu-Ende-Verzögerung hängen vom Datenvolumen und wesentlich vom Anwendungskontext ab [32]. Ein Dateitransfer benötigt keine reservierte Bandbreite und keine garantierte Ende-zu-Ende-Verzögerung; er sollte nur möglichst schnell erfolgen. Sind diese Daten beispielsweise aber der Status und die Koordinaten eines Zeigers auf einem gemeinsamen Fenster, dann bestehen Forderungen bezüglich der maximalen Ende-zu-Ende-Verzögerung. Hieraus läßt sich zusammen mit der charakteristischen Datenrate eine untere Schranke der benötigten Bandbreite ableiten.

2. **Einzelbilder und Graphiken:** Für die Übermittlung von unkomprimierten Einzelbildern ist die Korrektheit aller Bits im Allgemeinen nicht entscheidend. Ein verfälschter, einzelner Farbwert stört den optischen Eindruck nicht in einem Bild, das aus beispielsweise 1000 mal 1000 Bildpunkten aufgebaut ist. Geht hingegen eine ganze Zeile eines Bildes verloren, dann kann dies gegebenenfalls schon untragbar sein. Bei manchen Anwendungen, beispielsweise Röntgenaufnahmen in der Medizin, ist dagegen häufig jedoch jeder Bildpunkt relevant. Bei komprimierten Einzelbildern besitzen unterschiedliche Bits und Bytes in einem Datenstrom verschiedene Relevanz. So sind beispielsweise im *JPEG-Format nach einer sequentiellen Kompression die DCT-Koeffizienten mit den* niedrigsten zwei-dimensionalen Frequenzen sehr wichtig, während die DCT-Koeffizienten mit den höchsten Frequenzen relativ unwichtig sind. Der benötigte Bandbreitenbedarf sowie die maximal tolerierbare End-zu-End-Verzögerung sind wie bei Daten stark anwendungsabhängig. Für die zur Übertragung eines graphischen Objektes benötigte Bandbreite sind außerdem das gewählte Darstellungsverfahren, die Größe, Auflösung pro Pixel, Kodierung (beispielsweise 9-bit YUV) und Kompression wesentlich.

3. **Bildsequenzen:** Bei der Übermittlung von Bewegtbildern muß zwischen komprimierter und unkomprimierter Bildübertragung unterschieden werden. Die Zuverlässigkeitsanforderungen für unkomprimierte Bewegtbildübertragung sind geringer als für komprimierte Bewegtbildübertragung. Bei der unkomprimierten Bildübertragung sind TPDU-Verluste erträglich, da innerhalb von Sekundenbruchteilen das Folgebild den Fehler überlagert. Nicht so bei komprimierter Bewegtbildübertragung. Die meisten *Kompressionsverfahren (wie MPEG-1, DVI-PLV-Mode) basieren auf einer* Redundanzreduktion von zeitlich aufeinander folgenden Bildern. Hier folgt meist einem „interframe" komprimiertem Einzelbild (in MPEG das „I-Frame") mehrere „intraframe" komprimierte Bilder (in MPEG die „P und B-Frames"). Datenverlust und -verfälschungen wirken sind somit unterschiedlich aus. Der benötigte Durchsatz liegt bei komprimierter Bewegtbildübertragung im Bereich von 1 - 2 MBit/s, bei unkomprimierter Übertragung im Bereich bis zu 140 MBit/s (Bei HDTV-Qualität geht dies bis in den GBit/s-Bereich hinein). Da Bewegtbilder zeitkritische (kontinuierliche) Daten darstellen und es sich hier um eine Dialoganwendung handelt, ist eine maximale Übertragungsverzögerung von 600 ms unbedingt einzuhalten. Günstiger sind Werte um die 200 ms [32]. Auch die Schwankung der Übertragungsverzögerung „Jitter" ist zu beachten. Diese äußert sich insbesondere in den Speicherplatzanforderungen.

4. **Sprache:** Für Sprache gelten wegen der hier betrachteten Dialoganwendung und den zeitkritischen (kontinuierlichen) Daten die gleichen Anforderung bezüglich der maximalen Übertragungsverzögerung wie bei den Bewegtbildern. Die aus den maximalen Schwankungen der Übertragungsverzögerung abgeleiteten Speicherplatzanforderungen sind meist ca. um den Faktor 5-10 geringer als bei Bewegtbildern. Hier sind die erforderliche Qualität zusammen mit dem verwendeten Kompressionsalgorithmus die bestimmenden Faktoren. Der vom Benutzer akzeptierbare Jitter ist aber im Vergleich zur Bewegtbildübertragung geringer, weil das menschliche Ohr empfindlicher als das Auge ist. Der benötigte Durchsatz liegt ohne Kompression, in Abhängigkeit der Tonqualität, im Bereich von 64 kBit/s (Telefon) bis 173 kByte/s (Stereo in CD-Qualität).

5. **Medienmix:** Zwischen den Kommunikationspartnern (Transportdienstbenutzern) bestehen oft verschiedene Verkehrsströme für Daten, Einzelbild, Audio und Bewegtbilder.

Deshalb sollte das Kommunikationssystem eine Möglichkeit zur gemeinsamen Verwaltung solcher voneinander abhängiger Kommunikationsbeziehungen bieten.

An dieser Stelle sei nochmals explizit auf die Konsequenz der betrachteten interaktiven Kommunikationsbeziehung eingegangen: Es kommt nicht nur bei der Bewegtbild- und Sprachübertragung auf die Übertragungsverzögerung an, sondern auch bei den anderen Medien. So können beispielsweise die zu übertragenden Daten einen Mauspointer darstellen, der gleichzeitig, zu einem gesprochenen Kommentar, auf ein Detail in einer Graphik zeigen soll.

Im einem **zweiten Beispiel** sei ein **multimedialer Verteildienst** (Punkt-zu-Mehrpunkt) betrachtet. Hier sollen die Daten von einem Sender möglichst gleichzeitig zu verschiedenen Benutzern übertragen werden. Zusammenfassend lassen sich folgende, vom ersten Beispiel abweichende, Anforderungen ableiten:

- Die Übertragung der Daten zu den verschiedenen Benutzern sollte nicht durch mehrfaches Senden an die einzelnen Empfänger nachgebildet werden, da der Aufwand proportional mit der Anzahl der Empfänger steigt. Das Transportsystem sollte *Multicasting* unterstützen, damit mit einer einzigen TPDU möglichst viele Empfänger erreicht werden können.
- Beim *Multicast* müssen die Empfängergruppen festgelegt und verwaltet werden. Hierzu gehören alle Operationen, die den Status der Gruppe betreffen wie das Erzeugen und das Vernichten von *Multicast*-Gruppen, der Eintritt in die Gruppe und das Verlassen der Gruppe, Operationen im Zusammenhang mit Fehlererkennung und -behebung. Das Transportsystem sollte entsprechende Funktionalität enthalten oder einen extern verfügbaren *Directory*-Dienst verwenden.
- Das Transportsystem sollte dem Dienstbenutzer ermöglichen, einen Zuverlässigkeitsgrad entsprechend der Antwort-Semantik anzugeben (siehe [25]).
- Die maximale End-zu-End-Verzögerung ist wiederum anwendungsabhängig, es lassen sich jedoch meist höhere Werte im Bereich von maximal 1 Sek. tolerieren.

Klasse	Bitfehler	TSDU-Fehler
0	keine Sicherung	keine Sicherung
1	keine Sicherung	Sicherung mit Anzeige
2	Sicherung mit Anzeige	Sicherung mit Anzeige
3	keine Sicherung	Sicherung mit Behebung
4	Sicherung mit Behebung	Sicherung mit Behebung

Tabelle 1. Zuverlässigkeitsklassen zur Fehlerbehebung

Über die folgenden Dienste und Dienstgüte-Parameter können die Anforderungen an ein Transportsystem aus Sicht der multimedialen Anwendung befriedigt werden:
Neben der Punkt-zu-Punkt-Kommunikation ist eine Gruppen-Kommunikation notwendig. Der Dienstgüte-Parameter „Durchsatz" kann über die TSDU-Rate (TSDUs/s) zusammen mit einer mittleren Datenrate (Bytes/s) und maximalen TSDU-Größe (Bytes/TSDU) spezifiziert werden. Die maximale Übertragungsverzögerung wird mit dem *delay*-Parameter (ms) angegeben. Die Schwankung der Übertragungsverzögerung werden durch den Jitterparameter (ms) spezifiziert. Die Zuverlässigkeitsklasse wird über eine Klasse nach obiger Tabelle ([23]) ausgedrückt. Die Eckdaten des Gruppenmanagements sind über *max_member*, *min_member* und den Zuverlässigkeitsgrad anzugeben.

3. Xpress Transfer Protocol XTP

XTP [37] ist ein leichtgewichtiges, echtzeitfähiges Transferprotokoll und wurde als Teil des *Protocol Engine Projects* [13] bei der Firma *Protocol Engines Incorporated* entwickelt. Eines der wichtigsten Entwurfsziele von XTP war, eine Implementierung des Protokolls in VLSI-Technik zu ermöglichen bzw. zu erleichtern. Durch Zusammenfassung der Transport- und Vermittlungsschicht und Ausnutzung der hohen Geschwindigkeit und Parallelität moderner VLSI Implementierungen soll XTP in der Lage sein, die Datenübertragungsrate moderner Hochgeschwindigkeitsnetze Ende-zu-Ende zu unterstützen. Dieses Ziel soll erreicht werden, ohne einen Kompromiß bezüglich der Zuverlässigkeit und Funktionalität eingehen zu müssen.

Bestehende Protokolle wie TCP/IP oder ISO/TPx wurden nicht für Hochgeschwindigkeits-Netze entwickelt. Annahmen und Einschränkungen über Anwendungen und Netzcharakteristik, wie sie bei dem Entwurf dieser Protokolle gemacht wurden, treffen hier nicht mehr allgemein zu. Obwohl sie viele notwendige Funktionen wie Fehlererkennung, Flußregelung und Reihenfolgesicherung bereits enthalten, fehlen wichtige Mechanismen. Beispielsweise unterstützen sie keine *rate-control* oder selektive Übertragungswiederholung. Ein zuverlässiger *Multicast*-Dienst wird nicht angeboten. Das TPDU-Format ist komplex und erfordert umständliches *parsing* durch variable *Header*-Längen und Mehrfachbelegungen von Feldern. Das Erreichen eines hohen Parallelitätsgrades einer Implementierung ist aufgrund der verwendeten Protokollmechanismen nur bedingt möglich.

Neben der Möglichkeit, XTP in Hardware zu implementieren, werden in XTP Protokollmechanismen verwendet, die eine zuverlässige, realzeitähnliche Datenübertragung auch in Hochgeschwindigkeitsnetzen unterstützen. Unter realzeitähnlicher Verarbeitung wird nach der XTP Protokoll Definition die Fähigkeit verstanden, die Bearbeitungszeit für TPDUs in den Protokollinstanzen auf Sender- und Empfängerseite auf einen Zeitwert kleiner der TPDU-Übertragungsdauer zu begrenzen (bzw. NPDU-Übertragungsdauer bei Verwendung von XTP als Transferprotokoll - siehe [37], Seite 6).

Daneben bietet XTP einen *Multicast*-Dienst, sowie Fehler-, Fluß- und *Rate*-Kontrollmechanismen - ähnlich denen in anderen modernen Transportprotokollen[1]

Um die geforderte, realzeitähnliche Verarbeitung von PDUs zu ermöglichen wurde einerseits das PDU-Format optimiert, andererseits die Idee [12] aufgegriffen, die Funktionalität der Schichten 3 und 4 des ISO-OSI-Referenzmodells in einer Schicht, dem sog. *transfer layer* zusammenzufassen.

Durch diese Integration kann die Protokollbearbeitungszeit verkürzt werden, da in Transport- und Vermittlungsschicht redundant auftretende Funktionen wie beispielsweise Flußre-

[1] siehe beispielsweise TP + +, VMTP, NETBLT.

gelung und Fehlerbehandlung vermieden werden können. Diese Zusammenfassung von Schicht 3 und Schicht 4 dürfte im Umfeld von Weitverkehrsnetzen jedoch nur schwer zu realisieren sein, da hier die Funktionen der Vermittlungsschicht im Allgemeinen vom Netzbetreiber zur Verfügung gestellt werden. Aufgrund dieser Tatsache werden im folgenden primär Protokollfunktionen mit End-zu-Endsignifikanz (Transportprotokollfunktionen) untersucht.

4. TPDU-Format

Eine XTP TPDU besteht aus einem XTP *Header*, einem mittleren Segment und einem XTP *Trailer*. Das mittlere Segment besteht entweder aus einem Informationssegment oder aus einem Kontrollsegment. Die Syntax für den XTP*Header* und *Trailer* ist grundsätzlich für alle TPDUs gleich: 40 Byte *Header* und 4 Byte *Trailer*. Die Längen der Informations- bzw. Kontrollsegmente sind variabel. Die Länge jedes dieser Segmente - *Header*, *Trailer* und mittleres Segment - ist immer ein Vielfaches von 4 Bytes (*4 byte alignment*). Hierdurch ist eine besonders effiziente Implementierung auf Systemen mit 32-Bit Wortbreite möglich.

Der XTP Header enthält Informationen zur Steuerung folgender Vorgänge und Protokollfunktionen:

- Identifizierung von TPDUs (*key*)
- **Vermittlung von TPDUs durch ein** Internet (*route*)
- Begrenzung der Lebensdauer von TPDUs (*ttl*)
- Verwaltung der Reihenfolgenummern (*seq, dseq*)
- Durchführung von *Scheduling* (*sort*)
- Segmentierung (*dlen*)
- Fehlerbehandlung (*hcheck, sync*)

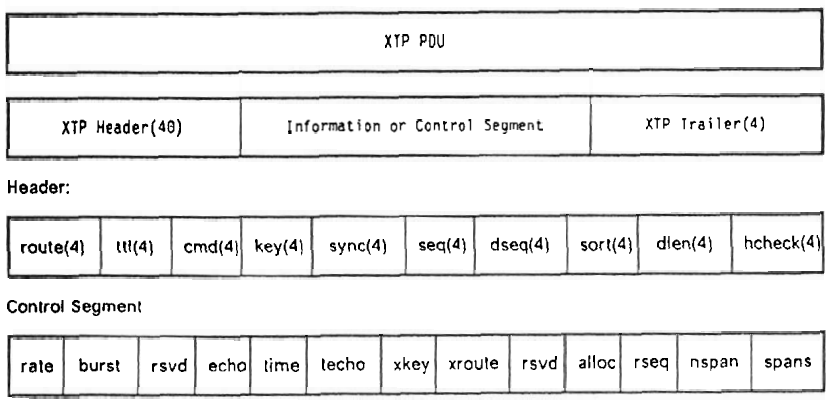

Abbildung 1. XTP-PDU Format

TPDUs, die ein Kontrollsegment beinhalten (sog. Kontroll-TPDUs) dienen dem Austausch von Zustandsinformation zwischen XTP-Protokollinstanzen.

Informationssegmente können sowohl Benutzerdaten eines Transportdienstbenutzers wie auch Protokolldaten (z.B. Adressegment oder Informationen zur Fehlerdiagnose) enthalten. In Informationssegmenten wird neben dem herkömmlichen Datenfeld zusätzlich ein *btag*-Feld (8 Byte) zur Verfügung gestellt, in dem Kontrollinformationen des Dienstbenutzers getrennt von den sonstigen Benutzerdaten übertragen werden können. Die Informationenen im *btag*-Feld werden von XTP wie die übrigen Benutzerdaten transparent übertragen.

Bei den hier betrachteten multimedialen Anwendungen könnte dieses *btag*-Feld vom Benutzer beispielsweise zur Synchronisation mehrerer voneinander abhängiger Kommunikationsbeziehungen verwendet werden. Ebenfalls wäre denkbar, daß der Dienstbenutzer das *btag*-Feld zur Übertragung von Zeitstempeln benutzt. Anhand solcher Zeitstempel könnte der Transportdienstbenutzer bei zeitkritischen Daten feststellen, ob die maximal zulässige Übertragungsdauer einer PDU überschritten wurde.

Aufgrund des Anforderungsprofils in Kapitel 2 ist jedoch für die Überwachung bzw. Einhaltung der Grenzwerte für Übertragungsverzögerung und Jitter das Transportsystem zuständig und nicht der Transportdienstbenutzer: Garantierte Grenzwerte für die End-zu-End-Übertragungsverzögerung können nur in Schicht 3 unterstützt werden, da nur hier die benötigten Informationen über die an der Übertragung beteiligten Teilnetze zur Verfügung stehen. So hat beispielsweise die Wegewahl (*routing*) großen Einfluß auf die Übertragungsverzögerung. Die Sicherstellung der geforderten Bandbreite kann ebenfalls nur in Schicht 3 erfolgen, da es nur hier möglich ist, die entsprechenden Netzressourcen in allen beteiligten Teilnetzen zu reservieren.

Neben der absoluten End-zu-End-Verzögerung spielt bei der Unterstützung multimedialer Anwendungen die Varianz der Verzögerung - auch als Jitter bezeichnet - eine wichtige Rolle. Transportsysteme sollten in der Lage sein, den Jitter durch geeignete Kompensationsmechanismen zu reduzieren. Jitter-Kompensation kann durch Zwischenspeichern der empfangenen Benutzerdaten erreicht werden. Hierzu wird zum einen ein Zwischenspeicher mit ausreichender Größe und zum anderen die Kenntnis über die aktuelle Verzögerung übertragener PDUs benötigt. Die Ermittlung der aktuellen End-zu-End-Übertragungsverzögerung kann grundsätzlich nur mit Hilfe von Zeitstempeln in den übertragenen TPDUs erfolgen.

Soll die Jitter-Kompensation in Schicht 4 erfolgen, müßte das XTP Protokoll um diese Funktion erweitert werden [1]: Das TPDU Format könnte um ein entsprechendes Feld ergänzt oder ein bereits vorhandenes Feld zweckentfremdet werden. Zur Zweckentfremdung bietet sich das bereites erwähnte *btag*-Feld an, das dem Dienstbenutzer dann nicht mehr zur Verfügung stehen würde oder alternativ das *sort*-Feld, das bei nicht gesetztem SORT-Bit bisher unbenutzt ist. Zusätzlich müßte XTP um eine Pufferverwaltung mit entsprechendem Timer-Mechanismus erweitert werden. Grundsätzlich könnte die Jitter-Kompensation auch in Schicht 3 erfolgen - jedoch sprechen folgende Argumente für eine Kompensation in Schicht 4:

Der für Jitter-Kompensation benötigte Speicherbedarf in Endsystemen wächst proportional mit der Anzahl der Transportverbindungen die eine Jitter-Kompensation benötigen. Wenn eine Kompensation nach jeder Teilstrecke erfolgt, wird zwar der benötigte Zwischenspeicher pro Netzverbindung in Schicht 3 kleiner sein (die zu glättenden Zeiten sind geringer), diese Einsparrung kann in einem Vermittlungsknoten (*Gateway*) jedoch schnell zunichte gemacht werden, wenn mehrere (z.B. hunderte) Netzverbindungen eine Jitter-Kompensation benötigen.

5. Verbindungsmanagement

Beginnend mit dem Verbindungsmanagement werden in den folgenden sechs Kapiteln die wichtigsten Aspekte eines Kommunikationsprotokolls am Beispiel XTP für Multimedia im Detail diskutiert.

Verbindungen werden in XTP grundsätzlich implizit [35] mit der Übertragung der ersten TPDU (sog. FIRST-TPDU) zwischen den beteiligten Transportprotokollinstanzen aufgebaut. Die FIRST-TPDU enthält ein Adreßsegment und optional zusätzlich ein Datensegment. Das Adreßsegment beinhaltet Adressierungsinformationen der Sender- und Empfängerinstanz. Es werden unterschiedliche Adressierungsformate, z.B. Internet -, ISO-, XNS-, IEEE 802 Source Route Address -, IP Source Routes Address - und XTP Direct (*locally-defined*) Address - Format unterstützt. Aufgabe des Adreßsegments ist neben der Adressierung von Sender- und Empfängerinstanz die Festlegung des Diensttyps, die Etablierung eines Pfades für Daten- und Kontroll-TPDUs sowie die Festlegung der vom Sender gewünschten Dienstgütewerte für *rate*, *burst* und *maxdata* (siehe Flußkontrolle und Flußsteuerung sowie Dienstgüte).

XTP erlaubt dem Dienstbenutzer, schon während der Verbindungsaufbauphase Daten zu übertragen. Das bedeutet, daß der Dienstbenutzer mit der Übertragung sofort beginnen kann, ohne zuvor auf eine Bestätigung der Verbindungsaufbauanforderung warten zu müssen. Hierdurch können zusätzliche Wartezeiten beim Verbindungsaufbau vermieden werden.

Nach dem Verbindungsaufbau erfolgt die Zuordnung einzelner TPDUs zu einer Verbindung mit Hilfe des *key*-Wertes im Header-Segment. Hierduch kann zum einen Bandbreite (Adress-Segment nur in der FIRST-TPDU enthalten) eingespart und zum anderen die erforderliche Verarbeitungszeit (*context lookup*) verkürzt werden.

Um die Bearbeitungszeit weiter zu verkürzen, sieht XTP beim Sender die Segmentierung von TSDUs in mehrere TPDUs vor. Der Sender kennt nach erfolgreicher Übertragung der FIRST-TPDU die maximale Größe (*maxdata*) einer TPDU die - ohne zusätzliche Segmentierung - durch alle, auf dem gesamten Pfad zur Empfängerseite beteiligten Netze unterstützt wird. Hierdurch wird eine sonst ggf. erforderliche mehrmalige Segmentierung an Netzwerkgrenzen vermieden.

Im Dienst-Feld kann vom Benutzer der geforderte Diensttyp festgelegt werden. Mögliche Diensttypen sind:

- Verbindungsorientierter Dienst
- Transaktion
- unbestätigtes Datagramm
- bestätigtes Datagramm
- isochroner Strom
- Massendaten

Durch die Festlegung des Diensttyps werden die an der Übertragung beteiligten Systemkomponenten über das zu erwartende Kommunikationsmuster informiert. Ensprechend können die benötigten Kommunikationsressourcen reserviert werden. Beispielsweise muß beim isochronen Strom garantiert werden, daß die angeforderte Bandbreite (ensprechend der *rate* und *burst* Werte im Adreß-Segment) als konstanter Wert anzusehen ist und nicht - wie z.B. beim normalen verbindungsorientierten Dienst möglich - im Überlastfall kurzzeitig reduziert werden kann.

Kann die Partnerinstanz die gewünschten Anforderungen nicht garantieren, wird dies dem Sender der FIRST-TPDU durch eine entsprechende Diagnose-TPDU (DIAG TPDU) angezeigt und der Verbindungsaufbau wird abgelehnt.

Die Festlegung des Diensttyps beim Verbindungsaufbau mit den entsprechenden Dienstgüteparametern (*rate-*, *burst-* und *maxdata*-Werte) eignet sich grundsätzlich auch für **multimediale** Anwendungen. Um eine praxisgerechte Aushandlung der Dienstgüte zu ermöglichen, sollte jedoch neben der Angabe der im ungünstigsten Fall noch akzeptierbaren Werte auch die Angabe der vom Dienstbenutzer gewünschten Dienstgütewerte möglich sein (z.B. *des_rate_req*, *des_burst_req*). Entspechend dem Anforderungsprofil in Kapitel 2 sollten zur Unterstützung multimedialer Anwendungen noch weitere Parameter eingeführt werden (z.B. *end to end delay* und *jitter*), um eine präzisere Spezifikation der geforderten Dienstgüte zuzulassen.

6. Prioritätsverfahren

XTP unterstützt ein prioritätsgesteuertes Planungsverfahren (*priority scheduling*) um die Bearbeitungsreihenfolge empfangener bzw. zu sendender TPDUs unterschiedlicher Verbindungen kontrollieren zu können. Zu diesem Zweck sieht XTP ein *sort*-Feld (32-Bit) im *Header* vor, das nur dann interpretiert wird, wenn das SORT-Bit (*cmd*-Feld im *Header*) gesetzt ist. Durch das *sort*-Feld werden 2^{32} unterschiedliche Prioritätsklassen festgelegt, wobei *sort* = 0 die höchste und *sort* = 2^{32} - 1 die niedrigste Klasse darstellt. TPDUs mit höherer Priorität werden vor TPDUs mit niedrigerer Priorität bearbeitet. TPDUs, in denen das SORT-Bit nicht gesetzt ist, werden erst nach TPDUs mit gesetztem SORT-Bit bearbeitet. Die Interpretation des *sort*-Feldes ist optional, XTP-Implementierungen müssen das Feld nicht zwingend bei der Bearbeitung berücksichtigen.

Bei der Verarbeitung der TPDUs sollte grundsätzlich nach 2 Klassen unterschieden werden:

1. **Daten mit Echtzeitanforderungen:** Bearbeitung gemäß eines Echtzeitplanungsverfahrens wie beispielsweise *earliest deadline first* oder *rate monotonic*.
2. **Daten ohne Echtzeitanforderungen:** Die nach der Verarbeitung übrigbleibende Verarbeitungskapazität soll für die Bearbeitung dieser Daten verwendet werden. Hier kann ein prioritätsgesteuertes Verfahren über das *sort*-Feld verwendet werden.

Die Wahl der Prioritätswerte erfolgt in XTP lokal bei der Senderinstanz. Die Abbildung der Dienstanforderungen auf die Prioritätswerte kann jedoch nur bei Kenntnis der Netzauslastung in allen an der Kommunikation beteiligten Teilnetze erfolgen und sollte daher in der Vermittlungsschicht erfolgen. In XTP sind die dazu erforderlichen Funktionen - auch bei Betrachtung des Gesamtfunktionsumfangs (Schicht 3 und 4) - nicht enthalten. Die Verwendung der Prioritätsmechanismen zur Sicherstellung der Echtzeitanforderungen *multimedialer* Anwendungen ist daher aus unserer Sicht nicht möglich.

7. Fehlerbehandlung

Wenn die Zuverlässigkeitsanforderungen der Transportdienstbenutzer nicht bereits durch den Vermittlungsdienst erfüllt werden, dann müssen in der Transportschicht Mechanismen zur Fehlererkennung, -meldung und -behebung bereitgestellt werden.

Fehlererkennung
Bitfehler werden in XTP mit Hilfe von zwei unterschiedlichen Prüfsummen erkannt. Eine der Summen wird über dem *Header*, die andere über dem mittleren Segment gebildet. Die *Header*-Prüfsumme wird grundsätzlich immer berechnet, die Berechnung der Prüfsumme über dem mittleren Segment kann mit Hilfe des NOCHECK-Bits im *cmd*-Feld des *Headers* kontrolliert werden.

Die Erkennung von **TPDU-Verlusten, Duplikaten oder Reihenfolgevertauschungen** erfolgt mit Hilfe von Reihenfolgenummern. XTP numeriert die übertragenen Datenbytes modulo 2^{32}. Das *seq*-Feld des *Header* enthält die Reihenfolgenummer des ersten Bytes im Daten-Segment. Zusammen mit dem *dlen*-Feld können die bisher empfangenen Benutzerdaten identifiziert und Lücken erkannt werden. Die Überwachung der Reihenfolgenummern kann durch das NOERR-Bit im *cmd*-Feld des Headers gesteuert werden.

Fehlermeldung
Mechanismen für die Fehlermeldung ermöglichen dem Empfänger, den Sender über erkannte Fehler zu unterrichten. XTP bietet hierzu zwei unterschiedliche Verfahren an: *go-back-n* und *selective reject*. Beim *go-back-n* Verfahren unterrichtet der Empfänger den Sender über die Reihenfolgenummer, von der an Daten verloren gingen oder vermißt werden. Detailliertere Informationen werden beim *selective reject* mitgeteilt. Hier werden alle vermißten Daten dem Sender durch Bereichsangaben (*gaps*) mitgeteilt.

Grundsätzlich fordert in XTP die Senderinstanz die Empfängerinstanz auf, Fehler zu melden. Dies geschieht durch Senden einer TPDU mit gesetztem *sreq*- oder *dreq*-Bit. Neben dieser Methode existiert noch das *fast-negative-acknowledgment* Verfahren (*FASTNAK*-Bit im *cmd*-Feld des *Headers* gesetzt). Hierbei informiert die Empfängerinstanz die Senderinstanz unmittelbar nach Erkennung eines Fehlers.

Fehlerbehebung
XTP verwendet zur Fehlerbehebung ausschließlich die Übertragungswiederholung. Dabei wird die Information über verlorene bzw. verfälscht empfangene Daten verwertet. Wenn der Empfänger nicht in der Lage ist, Daten, die außerhalb der Reihenfolge empfangen wurden, **zwischenzuspeichern, kann in XTP das *go-back-n*** Verfahren benützt werden. Hierbei werden alle Daten ab einer bestimmten Reihenfolgenummer nochmals **übertragen. Obwohl dieses** Verfahren die Protokollbearbeitung vereinfachen kann, führt es leicht zur Verschwendung von Bandbreite, insbesondere bei Netzen mit einer hohen Übertragungskapazität und einer großen Übertragungsverzögerung (z.B. Satelitennetzwerke) [20]. Neben diesem Verfahren bietet XTP auch noch die *selektive* Übertragungswiederholung an. Dabei werden nur diejenigen Daten nochmals gesendet, die der Empfänger nicht korrekt oder garnicht empfangen hat. Durch Tupel von Reihenfolgenummern wird dem Sender mitgeteilt, welche Daten korrekt empfangen wurden. Der Sender hat dann die Aufgabe, aufgrund dieser Information, diejenigen Daten zu bestimmen, die nochmals übertragen werden müssen. Dieses Verfahren spart Bandbreite ein, benötigt aber beim Empfänger große Zwischenspeicher und erhöht insgesamt die Komplexität des Protokolls.

Durch die beschriebenen Mechanismen eignet sich XTP prinzipiell auch zur Unterstützung multimedialer Anwendungen. Aufgrund seiner hohen Flexibilität können alle in Kapitel 2 aufgeführten Zuverlässigkeitsklassen unterstützt werden.

Da im allgemeinen die zu übertragenden Daten multimedialer Anwendungen zeitkritische Datenströme darstellen, kann jedoch unter Umständen das Verfahren der Fehlerbehebung durch Übertragungswiederholung nicht angewendet werden. Die maximale, vom Dienstbenutzer akzeptierte End-zu-End-Übertragungsverzögerung kann gegebenenfalls eine erneute Datenübertragung im Fehlerfall unmöglich machen. Zur Vermeidung dieses Problems bietet sich eine *forward error correction* Methode an. Hier können in Abhängigkeit der zu erwartenden Fehlercharakteristik (Bitfehler-, Bitbündelfehlerrate, TPDU Verlustrate) unterschiedliche Verfahren zum Einsatz kommen (z.B. das der *Cross Interleaved Reed Solomon Code* aus der CD-DA-Technologie oder andere fehlerkorrigierende Codes [6]). Alle diese Mechanismen verbrauchen jedoch zusätzliche Bandbreite und Rechenzeit.

Zusätzlich zu den genannten Fehlerarten sollten im Umfeld multimedialer Anwendungen auch Fehler durch verspätet eintreffende TSDUs erkannt und ggf. dem Benutzer angezeigt werden. Die hierzu erforderlichen Protokollerweiterungen wurden bereits im Abschnitt „TPDU-Format" beschrieben.

8. Flußregelung und Flußsteuerung

Flußregelung bzw. Flußsteuerung ist ein wichtiger Gesichtpunkt in Hochgeschwindigkeitsnetzen, da ein erhöhtes Risiko besteht, daß ein Empfänger bzw. ein Vermittlungsrechner durch die hohe Übertragungsgeschwindigkeit mit Daten überflutet wird. In diesem Zusammenhang wurde beobachtet, daß der Hauptgrund für Datenverluste in Hochgeschwindigkeitsnetzen eben in einer solchen Überlastung von Vermittlungsrechnern zu sehen ist.

XTP bietet zwei verschiedene, sich ergänzende Mechanismen zur Flußsteuerung. Dies ist zum einen das klassische Schiebefensterverfahren, in dem der Datenempfänger, ausgehend von seinem vorhandenen Empfangspufferplatz Senderechte explizit durch Übermittlung von Sequenznummern an den Sender vergibt.

Der zweite Mechanismus, in XTP als *rate control* bezeichnet, betrachtet nicht den Pufferplatz, sondern die Verarbeitungsleistung des Empfängers [byte/s] als kritische Größe (bei Einsatz als *Transfer*-Protokoll wird der im Netz realisierbare Durchsatz als weitere Größe hinzugenommen). Die protokollinterne Realisierung erfolgt über drei Variable *credit*, *burst* und *refresh-timer*. Ausgehend von einem zyklischen Verhalten, wie es im Multimediakontext meist anzutreffen ist, bezeichnet *refresh-timer* die Länge eines Zyklus [s], *burst* die maximale Menge der in einem Zyklus zu übertragenden Daten [byte] an. Die lokale Variable *credit* gibt zu jedem Zeitpunkt die bis zum nächsten Ablauf des *refresh-timer* noch übertragbare Datenmenge in Bytes an.

Die beiden Parameter *burst* und *refresh-timer* werden beim Verbindungsaufbau zwischen Sender und Empfänger ausgehandelt. Während der laufenden Übertragung wird die Einhaltung der Absprachen jedoch allein durch den Sender ohne Mitwirkung des Empfängers erzwungen[2].

---

[2] Erläuterung: Mit jeder übertragenen TPDU wird der Wert von *credit* um die Anzahl der in der TPDU enthaltenen Benutzerdaten verringert. Die Datenübertragung hält an, wenn der Wert von *credit* kleiner oder gleich Null wird. Nach jeder Periode des *refresh-timer* wird der Wert von *credit* wieder auf den ursprünglichen Wert von *burst* gesetzt. Ein Wert von Null für *burst* bedeutet, daß keine Flußsteuerung stattfindet, wohingegen ein

Wird XTP als *Transfer Layer Protocol* zur Erbringung von Vermittlungs- und Transportfunktionen eingesetzt, dürfen Vermittlungsrechner die Werte für *rate* und *burst* in Kontroll-TPDUs verändern. Zudem brauchen Vermittlungsrechner nicht auf eine passierende Kontroll-PDU zu warten, sondern können zu jeder Zeit eine sog. *Route-Control*-PDU (RCNTL-PDU) mit entsprechenden Werten für *rate* und *burst* ausgeben. Wenn ein Vermittlungsrechner *rate*- und *burst*-Werte in passierenden PDUs für sämtliche Verbindungen registriert, so ist er in der Lage, die Summe dieser Werte zu bilden. Diese Summe bestimmt den Bandbreitenbedarf des Vermittlungsrechnes. Durch Modifikation der Kontroll-PDUs und der Fähigkeit der Summenkontrolle, ist der Vermittlungsrechner in der Lage, Bandbreite zu verwalten.

Da die erforderliche Bandbreite und das zeitliche Verhalten der einzelnen zu übertragenden TSDUs in multimedialen Anwendungen vorgegeben sind, ist eine Flußregelung mittels Schiebefenster zwischen den Partnerinstanzen nicht sinnvoll anwendbar, sondern nur eine Flußsteuerung via *rate control*. In dieser Beziehung eignet sich XTP, da die Flußregelung ausgeschaltet werden kann („NOFLOW" Steuerfeld gesetzt) und eine Flußsteuerung protokollintern vorhanden ist. Für multimediale Anwendungen sollte jedoch auch eine Flußsteuerung am Dienstzugangspunkt vorgesehen werden, da ein „Wohlverhalten" des Dienstbenutzers am TSAP im allgemeinen nicht angenommen werden kann. Unter „Wohlverhalten" wird hier verstanden, daß ein Dienstbenutzer periodisch auftretende Daten dem Transportsystem stets auch absprachegemäß mit derselben Periode übergibt.

9. Gruppenkommunikation

XTP definiert einen *Multicast*-Modus, in dem ein Sender TPDUs gleichzeitig zu einer Gruppe von Empfängern überträgt (unidirektionales *one-to-many*).

Der *Multicast*modus gleicht in vieler Hinsicht dem Modus für einen einzelnen Empfänger. Der Sender gibt eine FIRST-TPDU und nachfolgend Daten-TPDUs aus. Eine Fehlerbehandlung ist vorgesehen. Für die Übertragungswiederholung wird das Verfahren *go-back-n* eingesetzt. Selektive Übertragungswiederholung wird nicht unterstützt. Da in der Menge der Empfänger die Größe der verfügbaren Empfangspuffer sehr unterschiedlich sein kann, wird die Datenübertragung mit der Verarbeitungsgeschwindigkeit des langsamsten Empfängers durchgeführt. Entdeckt ein Empfänger eine TPDU außerhalb der Reihenfolge, so wird eine Kontroll-TPDU (sog. *reject packet*) ausgegeben. Diese Kontroll-TPDU wird an die Menge aller an der Verbindung beteiligten Instanzen inklusive aller Empfänger übertragen, um sie über das Auftreten eines Fehlers zu informieren. Würden alle Fehler grundsätzlich von jedem Empfänger gemeldet bei dem sie erkannte wurden, ließe sich die Überschwemmung des Senders mit ankommenden Kontroll-TPDUs (*reject packets*) kaum vermeiden. Um diesen Effekt einzudämmen, dürfen die Empfänger keine Kontroll-TPDU senden, wenn sie davon ausgehen können, daß der Sender bereits über den Fehler informiert wurde. Während ein Empfänger eine Kontroll-TPDU erstellt und auf die Übertragung wartet, horcht er das Netz nach anderen Kontroll-TPDUs ab. Kommt eine andere Kontroll-TPDU für denselben Sender an, vergleicht der Empfänger seinen Wert für die Übertragungswiederholung mit dem Wert in der gerade empfangenen TPDU. Wird sein Wert durch den der TPDU abgedeckt, kann er seine eigene Kontroll-TPDU verwerfen (sog. *damping* Verfahren). Andernfalls schickt er seine Kontroll-TPDU zum Sender.

Wert von Null für *rate* die Datenübertragung anhält. Kontroll-TPDUs dürfen jedoch auch in diesem Fall weiter ausgegeben werden. Die Werte für *rate* und *burst* gelten jeweils für eine Richtung der Datenübertragung. So sind in Abhängigkeit der Übertragungsrichtung unterschiedliche Übertragungsraten möglich.

Neben diesem Verfahren bietet XTP auch eine unzuverlässige Datenübertragung im Multicast Modus. Bei diesem *no error* Modus verwirft ein Empfänger jene TPDUs die verfälscht sind. Er informiert lediglich den Dienstbenutzer über das Ereignis.

Aufgrund der oben beschriebenen Verfahren bietet XTP folgende Zuverlässigkeitsgrade für die Gruppenkommunikation [17]:

Zuverlässigkeitsgrad k = 0, die *Multicast*-Datenübertragung wird als erfolgreich angesehen, auch wenn kein Empfänger die TPDUs korrekt erhalten hat.

Zuverlässigkeitsgrad k = 1, die *Multicast*-Datenübertragung wird als erfolgreich angesehen, wenn mindestens 1 Empfänger die TPDUs erhalten hat

Die Fehlerkorrektur erfolgt durch wiederholtes Senden der verlorenen TPDUs an die gesamte *Multicast*-Gruppe.

Da in XTP keine Mechanismen für die Festlegung und Verwaltung von *Multicast*-Gruppen vorgesehen sind, ist eine Erweiterung des Zuverlässigkeitsgrads auf k = 2 bis k = Gruppengröße nicht ohne Änderung des Protokolls möglich [25]. XTP beinhaltet zwar einen Mechanismus, der es einem Empfänger ermöglicht, an einer bestehenden *Multicast*-Kommunikationsbeziehung teilzunehmen (*join group* Funktion). Das Eintreten eines weiteren Empfängers in die Gruppe wird jedoch nicht verwaltet. Zudem müssen beim Sender ankommende Kontroll-TPDUs eindeutig dem jeweiligen Initiator zugeordnet werden können (z.B. zur Feststellung ob ein Empfänger ausgefallen ist). Das bedeutet, daß das *damping* Verfahren nicht angewandt werden kann und, daß die Kontoll-TPDUs eine Identifikation des jeweiligen Initiators enthalten müssen. Zudem muß jede dynamische Veränderung der Gruppe dem Sender mitgeteilt werden.

Der von XTP erbrachte *Multicast*-Dienst eignet sich für reines *Broadcast* (z.B. Nachrichtenverteildienst, Zuverlässigkeitsgrad k = 0) und zur Suche einer bestimmten Information in einem verteilten System (z.B. finde einen freien Rechner, Zuverlässigkeitsgrad k = 1).

Ein großer Nachteil ist jedoch die nur unidirektionale *one-to-many* Kommunikationsbeziehung. XTP sollte neben den oben aufgeführten Erweiterungen auch bezüglich eines bidirektionalen *one-to-many Multicast* modifiziert werden.

10. Dienstgüteparameter

XTP unterstützt verschiedene Dienstgüteparameter, die nicht immer für die Multimedia-Kommunikation ausreichen. Im Folgenden seien die wichtigsten kurz aufgeführt:

Dienstgüte-Parameter	*Anmerkungen*
Verzögerung	In XTP werden nur Durchschnittswerte für die End-zu-End-Verzögerung festgelegt. Es sind keine Garantien bzw. Mechanismen vorgesehen um die angegebenen Wert einzuhalten. Dies ist somit nicht für die Kommunikation von Audio- und Videodaten geeignet. Man beachte hier insbesondere die Anfor-

derungen von Seiten eines Dialogdienstes (wie in Abschnitt 2 beschrieben).

Zuverlässigkeitsklasse Der Dienstbenutzer gibt beim Verbindungsaufbau eine Zuverlässigkeitsklasse an. Die entsprechende Fehlerbehandlung wird durchgeführt. (Siehe Abschnitt zur Fehlerbehandlung, in dem die Implikation bezüglich Multimedia beschrieben wurden).

TPDU Größe XTP kennt nach Übertragung der FIRST-PDU die maximale TPDU-Größe, die ohne zusätzliche Fragmentierung an Subnetzgrenzen übertragen werden kann. Hier sehen wir bisher keine Problematik im Kontext von Audio- und Videokommunikation.

impl. Verbindungsaufbau XTP ermöglicht dem Dienstbenutzer bereits während des Verbindungsaufbaus Benutzerdaten zu übertragen. Diese Daten sind nur bedingt für ein kontinuierliches Medium einzusetzen, weil die Dauer der Übertragungsverzögerung zwischen der ersten PDU und den folgenden stark variieren kann.

Übertragungsrate Der Dienstbenutzer gibt seine gewünschte Sendedatenrate (*rate*- und *burst*-Werte) an, XTP führt entsprechend Ratenkontrolle durch. (Anmerkungen hierzu siehe Abschnitt zur Flußkontrolle und Flußsteuerung.

Empfangspuffer Der Dienstbenutzer gibt Empfangspuffergröße an. XTP führt entsprechend Flußkontrolle auf Anwendungsebene durch (*Reservation*-Modus). Dies muß mit dem Buffermanagement des Multimedia-Systems geeignet zusammenarbeiten.

11. Ausblick

XTP eignet sich nach dem in diesem Bericht aufgezeigten Punkten in seiner jetzigen Form nur bedingt zur Kommunikation kontinuierlicher Daten. Als Beispiele für Problembereiche seien hier noch einmal die Überwachung der End-zu-End Verzögerung und des Jitters erwähnt. Als wünschenswerte, und ab einer gewissen Gruppengröße notwendige, Erweiterung sehen wir im Hinblick auf *Multicast* auch eine eigene Gruppenverwaltung oder eine geeignete Schnittstelle zu externen *Directory*-Diensten an.

Mit den in diesem Papier aufgezeigten Erweiterungen ist jedoch im wesentlichen eine multimediale Kommunikation möglich. Wegen des konkreten Mangels an einem globalen Reservierungsmechanismus für netzinterne Betriebsmittel, wie er beispielsweise bei ST II geboten wird, haben wir uns entschieden weiter ST II in der Vermittlungsschicht als grundlegendes Protokoll zu verwenden. An der Definition von XTP als reinem Transportprotokoll, das dann oberhalb ST II eingesetzt wird, wird zur Zeit am ENC gearbeitet.

Literaturverzeichnis

[1] *G. Anastasi, M. Conti, E.Gregori;* **TPR: A Transport Protocol for Real Time Services in an FDDI Environment;** In: M. Johnson (Hrsg.); Protocols for High-Speed Networks; Elsevier Science Publishers B.V. (North Holland), IFIP, 1990

[2] *S. R. Ahuja, J. Robert Ensor, David N. Horn;* **The Rapport Multimedia Conferencing System;** Proceedings of the Conference on Office Information Systems, Palo Alto CA, March 1988, pp 1-8.

[3] *David P. Anderson, Ralf Guido Herrtwich;* **Internet Communication with End-to-End Performance Guarantees;** Informatik-Fachberichte, no. 293, Springer Verlag, 1991.

[4] *Heinrich Armbrüster, Hans Jörg Rothamel;* **Breitbandanwendungen und -dienste: Qualitative und quantitative Anforderungen an künftige Netze;** ntz, vol. 43, no. 3, 1990, pp. 150-159.

[5] *Bell Telephone Labs.;* **Engineering and Operations in the Bell System;** AT&T Bell Labs, 1984.

[6] *Ernst W. Biersack;* **Performance Evaluation of Forward Error Correction in ATM** Computer Communication Review, ACM SIGCOMM, Vol.22, No.1, Januar 1992.

[7] *Steward Brand;* **The Media Lab, Inventing the Future at MIT;** Viking Penguin, 1987.

[8] *Timothy S. Balraj, Yechiam Yemini;* **PROMPT - A Destination Oriented Protocol for High Speed Networks;** M. Johnson (Hrsg.); Protocols for High-Speed Networks; Elsevier Science Publishers B.V. (North Holland), IFIP, 1990

[9] *E.W. Biersack, C.J. Cotton, D.C. Feldmeier, A.J. McAuley;* **An Overview of the TP++ Transport Protocol Project;** submitted for publication

[10] *Ernst W. Biersack;* **Connection Management using Synchronized Clocks;** Proc. of Third IFIP WG 6.4 Conference on High Speed Networking, Berlin, March 1991, pp.225-238

[11] *International Telegraph and Telephone Consultative Committee;* **B-ISDN Service Aspects;** CCITT Study Group XVIII, Draft Recommendation I.211, Geneva, 23-25 May 1990.

[12] *CELAR (Centre d'Electronique de L'Armement);* **Military real time local area network;** Rennes, France, Februar 1987

[13] *Greg Chesson;* **The Protocol Engine Project;** UNIX Review, Vol.5, No.9, September 1987, pp.70-77

[14] *Greg Chesson;* **The Evolution of XTP;** Proc. of Third IFIP WG 6.4 Conference on High Speed Networking, Berlin, March 1991, pp.15-24.

[15] *David R. Cheriton, Erik Nordmark;* **Experiences from VMTP: How to achieve low response time;** H. Rudin, R. Williamson (Hrsg.); Protocols for High-Speed Networks; Elsevier Science Publishers B.V. (North Holland), IFIP, 1989, pp.43-56

[16] *David D. Clark, Mark L. Lambert, Lixia Zhang;* **NETBLT: A High Throuput Transport Protocol;** Proc. SIGCOMM'87, ACM, August 1987, pp.353-359

[17] *J. Crowcroft, K. Paliwoda;* **A Multicast Transport Protocol;** Trans. SIGCOMM, ACM, August 1988, pp.247-256

[18] *S. Deering;* **Host Extensions for IP Multicasting** Network Working Group, RFC 1112, August 1989

[19] *Willibald Doeringer, Doug Dykeman, Matthias Kaiserswerth, Bernd Meister, Harry Rudin, Robin Williamson* **A Survey of Light-Weight Transport Protocols for High-Speed Networks** IEEE Transactions on Communications, Vol.38, No.11, November 1990, pp.2025-2039

[20] *David C. Feldmeier, Ernst W. Biersack;* **Comparison of Error Control Protocols for High Bandwith-Delay Product Networks** In: M. Johnson (Hrsg.); Protocols for High-Speed Networks; Elsevier Science Publishers B.V. (North Holland), IFIP, 1990

[21] *Domenico Ferrari;* **Client Requirements for Real-Time Communication Services;** International Computer Science Institute, Technical Report 90-007, Berkeley, March 1990.

[22] *G. D. Flinchbaugh, P. L. Martinez, D. S. Rouse;* **Network Capabilities in Support of Multimedia Applications;** IEEE Global Telecommunications Conference & Exhibition (GLOBECOM), Conference Record vol. 1, San Diego, Dec. 2-5, 1990, pp. 322-326.

[23] *Lutz Henckel, Heiner Stüttgen;* **Transportdienste in Breitbandnetzen;** E. Effelsberg, H.W. Meuer, G. Müller (Hrsg.); Proceedings zur GI/ITG Fachtagung "Kommunikation in verteilten Diensten" in Mannheim; Springer Verlag; Februar 1991; pp.96-111

[24] *Dietmar Hehmann, Michael Salmony, Heinrich Stüttgen;* **Transport Services for Multimedia Applications;** Proceedings of the IFIP WG 6.1/WG 6.4 Workshop on Protocols for High Speed Networks, North Holland, 1989, 303-321

[25] *Larry Hughes;* **Multicast Response Handling Taxonomy;** Computer Communications, vol. 12 No 1, Feb. 1989, pp. 39-46.

[26] *Information Processing Systems - Open Systems Interconnection* **Transport service definition;** ISO 8072, 1986

[27] *Information Processing Systems - Open Systems Interconnection* **Transport protocol specification;** ISO 8073, 1986

[28] *Mark G.W. Jones, Soren-Aksel Sorensen, Steve R. Wilbur;* **Protocol design for large group multicasting: the message distribution protocol;** Computer Communications, Vol.14, No.5, June 1991, pp.287-297

[29] *Robert M. Sanders, Alfred C. Weaver;* **The Xpress Transfer Protocol (XTP) - A Tutorial** ACM Computer Communications Review, Vol.20, No.5, October 1990, pp.67-88

[30] *Thomas Schütt, Manny Farber;* **The Heidelberg High Speed Transport System: First Performance Results;** In: 3rd International IFIP WG6.1/6.4 Workshop on Protocols for High-Speed Networks, Stockholm, May 13-15, 1992 pp. 35-50.

[31] *Ralf Steinmetz,* **Multimedia Synchronization Techniques: Experiences Based on Different System Structures;** IEEE Multimedia Workshop '92, Monterey, CA, USA, April, 1992.

[32] *Ralf Steinmetz, Thomas Meyer;* **Modelling Distributed Multimedia Applications;** IEEE Int. WS on Advanced Communications and Applications for HS Networks, München, March 1992.

[33] *Claudio Topolcic;* **Experimental Internet Stream Protocol, Version 2 (ST-II);** RFC 1190, Oct. 1990.

[34] *R.W. Watson;* **Timer-Based Mechanisms in Reliable Transport Protocol Connection Management** Computer Networks, No.5, 1981, pp.47-56

[35] *R.W. Watson;* **The Delta-t Transport Protocol: Features and Experience;** H. Rudin, R. Williamson (Hrsg.); Protocols for High-Speed Networks; Elsevier Science Publishers B.V. (North Holland), IFIP, 1989, pp.3-18

[36] *David J. Wright, Michael To;* **Telecommunication Applications of the 1990s and their Transport Requirements;** IEEE Network Magazine, vol.4, no.2, March 1990, pp.34-40.

[37] *Protocol Engines Incorporated;* **XTP Protocol Definition Revision 3.6;** PEI 92-10, Protocol Engines Incorporated, Santa Barbara, CA 93101, Januar 92

Formale Beschreibungstechniken für Kommunikationsprotokolle: Probleme ihrer praktischen Anwendung

Hartmut König
Institut für Rechnerverbund und Betriebssysteme

Peter Neumann
Institut für Automatisierungstechnik

TU "Otto von Guericke" Magdeburg
Postfach 4120
O-3010 Magdeburg

Zusammenfassung

Formale Beschreibungstechniken für Kommunikationsprotokolle sind ein wichtiges und attraktives Forschungsgebiet im Umfeld der Kommunikationssysteme. In den letzten 10 Jahren sind viele markante Forschungsergebnisse erzielt worden. Die wichtigsten Techniken Estelle, LOTOS und SDL wurden standardisiert. Trotz der unbestrittenen Vorteile formaler Beschreibungstechniken stagniert ihre breite praktische Anwendung. Im vorliegenden Beitrag sollen die Gründe für diese Situation diskutiert und Schlußfolgerungen für eine breite praktische Anwendung gezogen werden. Anhand eines Fallbeispiels - der Formalisierung der deutschen Feldbusnorm PROFIBUS - soll über Erfahrungen bei der Anfertigung formaler Beschreibungen und ihrer Nutzung bei der (semi-) automatischen Implementierung und zum Konformitätstest berichtet werden.

1. Einleitung

Formale Beschreibungstechniken (FDTs) für Kommunikationsprotokolle sind ein wichtiges und attraktives Forschungsgebiet im Umfeld der Kommunikationssysteme. Die vielen Veröffentlichungen zu diesem Gebiet, vor allem im Rahmen der Konferenzen PSTV und Forte, sowie der Zuspruch zu diesen Konferenzen bzw. zu diesbezüglichen Sektionen in anderen Tagungen, belegen dies. Im Rahmen der Forschungsarbeiten zu formalen Beschreibungstechniken wurde ein breites Spektrum bemerkenswerter Forschungsergebnisse erzielt. Die markantesten Meilensteine sind die Standardisierung der Sprachen Estelle /ISO 9074/, LOTOS /ISO 8807/, SDL /CCITT 89/ und ASN.1 /ISO 8824/ sowie die Ausarbeitung der Standards zur Konformitätstestung. Daneben wurden eine Reihe von inhouse-Lösungen realisiert und erfolgreich eingesetzt /Holzmann 88/, /König 90/, /Schneider 92/.

Die Vorteile der Verwendung formaler Beschreibungstechniken sind unumstritten. Als Hauptvorteile gelten:
- Entwicklung eindeutiger, klarer und präziser Spezifikationen
- Möglichkeit der rechnergestützten Bearbeitung.

Sie begegnen damit dem generellen Nachteil der bis heute gültigen Standardisierungspraxis für Kommunikationsprotokolle, die auf der Verwendung verbaler Spezifikationen ergänzt durch Graphiken und Zustandstabellen beruht. Solche Spezifikationen lassen mehrdeutige Interpretationen des Textes zu, was zu Fehlern, unerwartetem und unerwünschtem Protokollverhalten und nichtadäquaten Implementierungen führen kann. Sie gestatten keine rechnergestützte Bearbeitung. Obwohl in der Praxis auf der Grundlage dieser Spezifikationen viele Protokolle erfolgreich implementiert worden sind, zeigt die Erfahrung, daß die Zahl der Fehler und Inkompatibilitäten unnötig hoch ist /Bochmann 90a/.

* Das diesem Beitrag zugrundeliegende Vorhaben wurde mit Mitteln des Bundesministers für Forschung und Technologie unter dem Förderzeichen 13I10090 gefördert.

N. Gerner H.-G. Hegering J. Swoboda (Hrsg.)

Kommunikation in Verteilten Systemen

ITG/GI-Fachtagung
München, 3.–5. März 1993

Springer-Verlag
Berlin Heidelberg New York
London Paris Tokyo
Hong Kong Barcelona
Budapest

Herausgeber

Nina Gerner
Siemens-Nixdorf Informationssysteme AG
Otto-Hahn-Ring 6, W-8000 München 83

Heinz-Gerhard Hegering
Leibniz-Rechenzentrum München
Barerstraße 21, W-8000 München 2

Joachim Swoboda
Lehrstuhl für Datenverarbeitung
Technische Universität München
Arcisstraße 21, W-8000 München 2

CR Subject Classification (1992): A.0

ISBN 3-540-56482-9 Springer-Verlag Berlin Heidelberg New York
ISBN 0-387-56482-9 Springer-Verlag New York Berlin Heidelberg

Satz: Reproduktionsfertige Vorlage vom Autor/Herausgeber
Druck- u. Bindearbeiten: Weihert-Druck GmbH, Darmstadt

Vorwort

Der Rechner am Arbeitsplatz ist bereits weitgehend eine Selbstverständlichkeit. Durch Vernetzung und zunehmend multimediale Eigenschaften entwickeln sich verteilte Systeme von Rechnern zu einer breit nutzbaren Infrastruktur der Informations- und Kommunikationstechnik. Die Verschmelzung dieser Gebiete, traditionell der Datenverarbeitung und der Fernmeldetechnik, hat sich in der Reihe der Tagungen "Kommunikation in Verteilten Systemen (KiVS)" schon frühzeitig angekündigt - und zu einer rasanten Entwicklung geführt. So ist es sicher keine Zufälligkeit, daß die KiVS auf dem Gebiet der Kommunikationstechnik zu einer der größten wissenschaftlichen Fachtagungen im nationalen Bereich heranwuchs.

Die KiVS'93 ist die 8. Tagung in ihrer Reihe und findet nach Berlin, Aachen, Stuttgart und Mannheim erstmals in München statt. Die Tagungen werden von dem Fachausschuß "Kommunikation und Verteilte Systeme" durchgeführt. Der Fachausschuß gehört sowohl der GI (Gesellschaft für Informatik) als auch der ITG an (Informationstechnische Gesellschaft im Verband Deutscher Elektrotechniker, VDE). Für die Durchführung der Veranstaltung in München ist die ITG verantwortlich.

Auf dem Gebiet der Kommunikation in Verteilten Systemen nehmen die Teilgebiete der Multimedia-Kommunikation und des Netzmanagements eine Schlüsselrolle in der Aktualität ein. Multimedia-Kommunikation steht in Wechselwirkung mit zahlreichen anderen Teilgebieten wie Hochgeschwindigkeitsnetzen und -protokollen, der Synchronisation der einzelnen Medien oder geeigneten Referenzmodellen multimedialer Kommunikation. Für das Netzmanagement stehen insbesondere Fragen einer integrierten Lösung für heterogene Netze an.

Die Fachtagung soll Gelegenheit zu einem konstruktiven Dialog zwischen Forschern, Entwicklern, Planern und Anwendern aus Universitäten, Forschungseinrichtungen, Industrie, Netzverwaltung und -betrieb bieten. Das Tagungsprogramm, dessen schriftliche Beiträge in diesem Tagungsband vorliegen, umfaßt wesentliche Teilgebiete wie

- Anwendungen in Büro, Fertigung und Anlagen
- Netzdienste und Wissenschaftsnetze
- Private und Intelligente Netze
- Integriertes Management heterogener Netze
- Multimedia- und Hochgeschwindigkeitskommunikation
- Architektur Verteilter Systeme
- Kommunikationsprotokolle und formale Beschreibungstechniken
- Leistungsanalyse.