1-1-1991

Comparative Study Of The VMTP, XTP, and TP4 Protocols And Their Functionalities From The Perspective Of The OSI Reference Model: Investigation Of OSI Protocols For Distributed Interactive Simulation

David T. Shen

INSTITUTE FOR SIMULATION AND TRAINING

Contract Number N61339-91-C-0103
October 31, 1991

Prepared for U.S. Army Project Manager for Training Devices

A Comparative Study of the VMTP, XTP and TP4 Protocols and their Functionalities from the Perspective of the OSI Reference Model

Investigation of OSI Protocols for Distributed Interactive Simulation

Institute for Simulation and Training
12424 Research Parkway, Suite 300
Orlando FL 32826

University of Central Florida
Division of Sponsored Research

IST-TR-91-26

INSTITUTE FOR SIMULATION AND TRAINING

A Comparative Study of the VMTP, XTP and TP4 Protocols and their Functionalities from the Perspective of the OSI Reference Model

Investigation of OSI Protocols for Distributed Interactive Simulation

Contract Number N61339-91-C-0103
October 31, 1991

Prepared for U.S. Army Project Manager for Training Devices
12350 Research Parkway • Orlando, FL 32826-3276

IST-TR-91-26

Written by

David T. Shen

Reviewed by

Michael Craft

Institute for Simulation and Training • 12424 Research Parkway, Suite 300 • Orlando, Florida 32826

University of Central Florida • Division of Sponsored Research

A COMPARATIVE STUDY OF THE VMTP, XTP AND TP4 PROTOCOLS AND THEIR FUNCTIONALITIES FROM THE PERSPECTIVE OF THE OSI REFERENCE MODEL

PREPARED FOR:

U.S. ARMY PROJECT MANAGER FOR TRAINING DEVICES
12350 Research Parkway
Orlando, Florida 32826-3276

INVESTIGATION OF OSI PROTOCOLS FOR
DISTRIBUTED INTERACTIVE SIMULATION

CONTRACT N61339-91-C-0103

October 31, 1991

Institute for Simulation and Training
University of Central Florida
12424 Research Parkway
Orlando, Florida 32826

ABSTRACT

This paper presents a comparative study of three communication protocols at the Transport layer of the Communication Protocol Stack; namely, the Versatile Message Transaction Protocol (VMTP), the Xpress Transfer Protocol (XTP), and the Transport Protocol Class 4 (TP4). This study focuses on the similarities and differences of these three protocols in terms of Transport layer functions, such as Connection Establishment, Checksum, Error Recovery, and Flow Control. The Transport layer functions defined in the Open System Interconnection Reference Model (OSIRM) will be used as the common basis for the comparison.

This study is motivated by the fact that older (i.e., in existence over 10 years) Transport layer protocols , such as the TP4, are not able to support a variety of today's applications requiring real-time network services, which the newer (i.e., in existence less than 5 years) Transport layer protocols, such as the VMTP and the XTP, are able to support. The fact that older protocols cannot support real-time applications has been demonstrated through experiences gained using these protocols[1]. Therefore newer protocols were designed to support the real-time applications and to provide new functionalities (e.g., multicast, message forward) not provided by older protocols. This study identifies some of the new concepts (e.g., optional acknowledgement, selective retransmission) used in the design of the VMTP and the XTP compared to those used in the design of the TP4, and

---

[1] Older protocols, such as TP4, are handicapped by their assumption of an unreliable underlying network.

examines how these new concepts in Transport protocol design can support today's application requirements.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

The proliferation of computers and other signal processing elements throughout the world, coupled with the need for reliable and efficient exchange of information, has driven the development of a number of computer networks. The differences in both requirements and environments for these networks have resulted in a variety of network designs. Furthermore, differences in requirements, coupled with changing technologies, have resulted in the manufacture of many different types of computers. These computers, although located in different networks, still need to interact.

Beginning in 1968 with the Advanced Research Project Agency Network (ARPANET) , the Defense Advanced Research Projects Agency (DARPA) has sponsored the development of a number of computer network technologies. This research has resulted in an architecture and a set of protocols to accomplish a robust system of interconnected networks.

The development of communication protocols to support interaction among computers located in different countries began in 1977 with the establishment of a subcommittee within the International Organization for Standardization (ISO). ISO subcommittee 16 was established to coordinate development of the Open System Interconnection Reference Model (OSIRM)..

OSI is a set of protocol standards built using the communication protocol layering model. The OSIRM is designed in seven layers: Application,

Presentation, Session, Transport, Network, Data Link, and Physical. These layers can be used to accomplish an open system architecture for a worldwide computer communication.

In the past five years many new Transport layer protocols (which deal with end-to-end data transmission) have been designed to incorporate new Transport layer features. These new Transport layer protocols promise better utilization of facilities provided by the network, making computer communication faster and more efficient. This study compares an earlier transport layer protocol with new transport layer protocols, and identifies the reasons for their better performance (e.g., increased throughput, decreased latency).

Two new protocols, the Versatile Message Transaction Protocol and Xpress Transfer Protocol will be compared with the older Transport Protocol Class 4. These three protocols are compared under a subset of functions defined in the OSI Transport layer protocol defined by ISO [7]. The OSI Transport layer protocol specification is used as the baseline in this study because it is an international standard, and it specifies what an open transport layer protocol should provide.

Computer Communications

When heterogeneous computer systems wish to communicate, two solutions are available:

1) one of the communicating systems needs to adhere to the communication rules of the other system, or

2) the computer systems must agree to use a common set of rules, possibly different from the ones normally used by either of the communicating parties.

Until recently, most computer communication problems have been solved through the first method, in which one system adheres to the requirements of the other system through specialized interfaces. The alternative is to use a shared set of protocols such as the ones provided by OSI. As a system-independent communication architecture, OSI allows computers to exchange information effectively, regardless of their native communications protocol.

Open Systems Interconnection (OSI)

History

OSI design was initiated in March 1977 by the ISO. Interest and participation have continued to grow. The objectives of OSI are [13]:

- Interconnection of different vendor's systems.
- Coordination of standardization activities and processes in telecommunications and information systems.
- Promotion of new information system business.

OSI standards are a major activity, not only in ISO, but also in the major national and international standards organizations. These include the American National Standards Institute (ANSI), the International Telegraph and Telephone Consultative Committee (CCITT), the European Computer Manufacturers Association (ECMA), and the Institute for Electrical and Electronics Engineers (IEEE).

The first plenary session of the Subcommittee 16[1] took place in February 1978 in Washington, D.C.. The result of this meeting was the first draft of the seven layer architectural Reference Model.

OSI has grown in technical concept from the development of a telecommunications base to the current scope of effort. The telecommunications

---

[1] SC16 was established by ISO to develop an architecture that would form the basis for the further development of a set of inter-system standards

base has expanded to incorporate technical innovations such as Local Area Networks (LANs), and Fiber Distributed Data Interface (FDDI) [13].

The OSI reference model reached the International Standard stage in 1983 with the seven layer protocol stack shown on Figure 1.

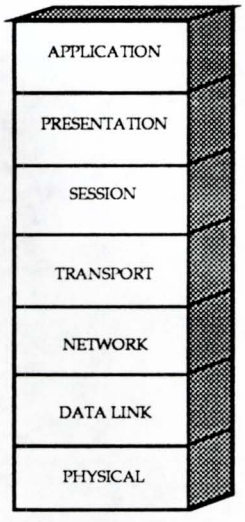

Figure 1. Open System Interconnection Basic Reference Model

The standardization effort has since increased with commitments from governments and industries to support adoption of OSIRM as the international standard protocol architecture and to promote its implementation.

Description

The wide variety of computer hardware and software complicates computer communication. One effective way to solve this problem is to divide the communication problem into several smaller tasks, solving the smaller problems separately. If the solutions to all the small problems work together, then the original problem is thereby solved.

This is the approach taken by OSI Reference Model and, in fact, by most communications architectures. The total problem is partitioned into sub-problems, and in the context of OSI Reference Model, these sub-problems are represented by layers [7].

The OSIRM is composed of seven layers. Each layer is assigned a specific group of the functions required to allow users of the communication services to exchange information. Each layer uses the layer's protocol to communicate with its peer. Vertical communication between layers is through the Service Access Points (SAPs).

The OSI communications architecture is organized as a protocol stack, i.e., there is a certain order in which the communication data is to be handled by the protocols. Each protocol performs its functions, solving its part of the problem, while relying on the layers below it in the stack to solve theirs. Each layer in the stack provides a set of services to the layer above it and uses the services provided by the layer below it.

OSI defines both an architecture for computer communications and protocols which fit within that architecture. As a set of protocols, it provides concrete rules for the communicating devices to interact. The rules, in principal, allow computer systems to communicate. As an architecture, it defines a structure for organizing the protocols, along with a set of terms and concepts for describing them. In order for two computer systems to be able to communicate, they must adhere to the same communication protocol architecture. It is not required that both systems implement protocols in the same way. As long as each follows the rules set by the protocols, communication may take place.

Protocol Data Units (PDUs) are the units of message passed from layer to layer within a protocol stack and transmitted across a computer network.

The OSI layers are [14]:

- Layer 1 - Physical

This layer provides the mechanical, electrical, functional, and procedural means to activate, maintain, and deactivate a physical link between systems.

- Layer 2 - Data Link

The data link layer deals with the requirements of the communications medium. The primary purpose of this layer is to provide a reliable means to transmit data across a single physical link by determining how the bits of the physical layer are grouped into frames, handling transmission errors, and regulating the flow of frames so receivers are not overwhelmed by faster senders.

In a Local Area Network (LAN) environment, this layer is divided into two sublayers: the Medium Access Control (MAC) sublayer and the Logical Link Control (LLC) sublayer. MAC deals with the data communication access to the physical medium. LLC provides the framework for data to be sent.

There are numerous types of protocols in the MAC sublayer such as CSMA/CD, Token Ring, Token Bus, etc.. These protocols are part of the OSI standard and are implemented by various vendors (e.g., 3COM, IBM).

- Layer 3 - Network

The network layer provides the means to establish, maintain, and terminate Network Connection (NC) and to exchange network service data units (NSDU) between transport entities. It provides the transport entities with independence from the details of the intervening sub-network consideration of routing and relaying functions required to transmit data units across sub-networks.

One very popular network layer protocol is the Internetwork Protocol (IP). There are two versions of IP, one designed by DARPA, termed Internet IP, and one designed by ISO, termed OSI IP.

- Layer 4 - Transport

The transport layer can be thought of as the highest of the lower layer protocols. Transport layer and below are concerned with the transmission of data between systems across a communications facility. The transport layer is characterized as providing transparent transfer of data between Session entities.

All protocols at this layer have end-to-end significance between transport entities in end systems.

- Layer 5 - Session

The purpose of the Session layer is to provide the means for cooperating Presentation entities to organize and synchronize their dialogue and to manage the data exchange.

- Layer 6 - Presentation

The Presentation layer deals with the representation of information of concern to Application entities (machine representations of data, encoding and the like). It handles the data to be transmitted between application entities (end-user information).

- Layer 7 - Application

The Application layer is the top layer between the open systems environment and the application processes that use the environment to exchange data. It is the Application layer that uses lower layer communication services to support user application processes. The Application layer does not provide services to a higher layer, instead these services are provided to application processes outside the seven layer OSI architecture.

OSI Transport Protocol Functionalities

Introduction

The Transport protocol is the cornerstone of the computer communications architecture over a Wide Area Network (WAN). It is the lowest layer supplying end to end connections. The Transport protocol is invisible to the network connecting the two communicating end systems (e.g., system A and system B, as shown in Figure 2 and 3).

In OSIRM terminology, the end-systems are called the Initiator and the Responder, indicating the system that initiates communication and its counterpart, respectively. The systems between the end-systems are termed intermediate systems and they convey messages to the end-systems.

The Transport protocol typically ensures that data is delivered error-free, in sequence, with no loss or duplication. On top of these basic requirements, it must meet the performance requirements dictated by a Quality of Service parameter, which expresses user requirements for the network services.

The Transport layer shields applications from the details of the underlying communications services. For each application (e.g., file transfer), an application-oriented protocol is needed to coordinate the activities of the corresponding application modules and ensure common syntax and semantics. The Transport protocol, in turn, makes use of network service modules, which provide access to the intervening communications network.

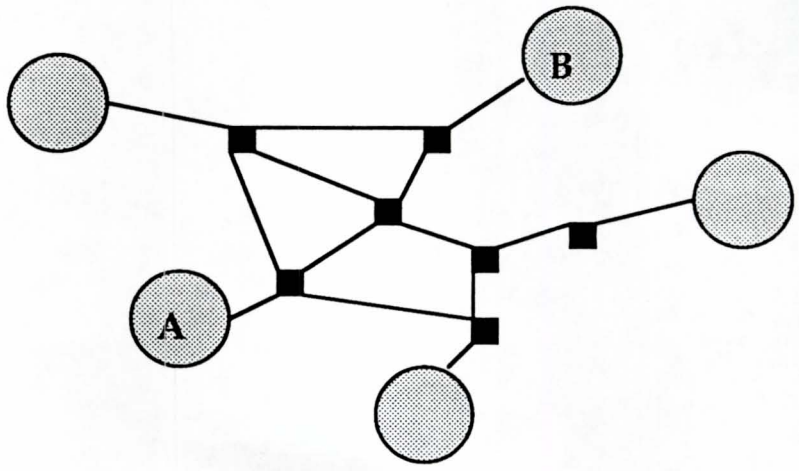

Figure 2. A Network of End Systems and Subnetworks

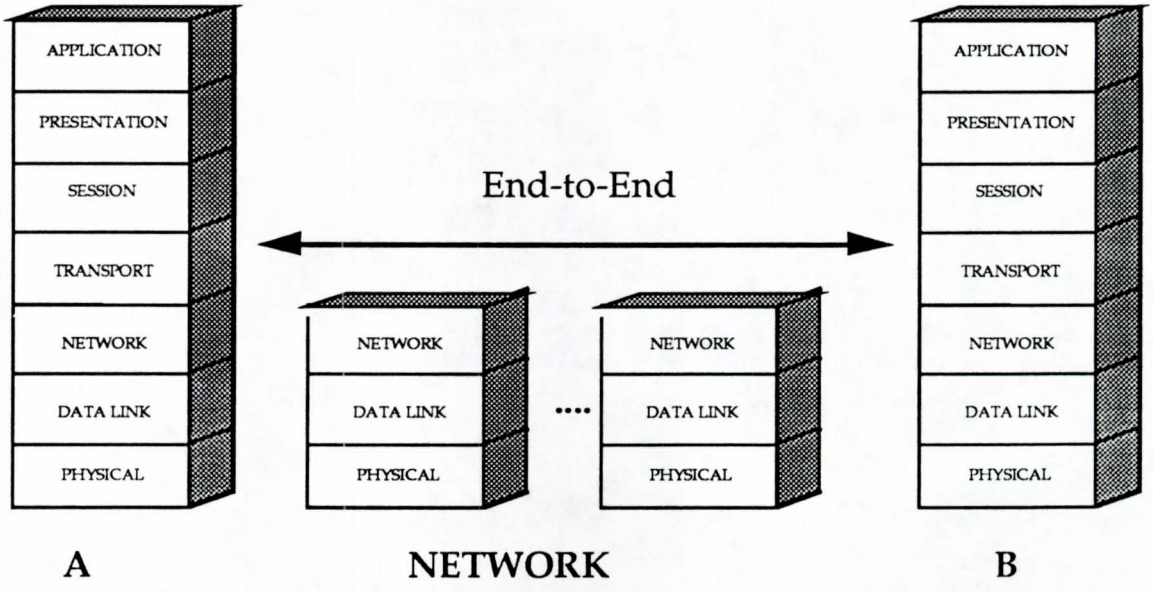

Figure 3. Transport Layer Protocol Providing End-to-End Communication Utilizing Intermediate Subnetworks

Classification

There are two modes of operation for Transport protocols: connection-oriented (CO) and connectionless (CL). In the CO mode, an initial connection set-up (handshake) between two or more hosts intending to establish communication takes place before the transmission of user data. During the connection set-up phase, the computers involved in the connection establishment negotiate the transport service and allocate local buffers and bandwidth for that specific connection. OSI defines 10 different types of Transport Protocol Data Units (TPDUs) for CO mode of communication [7].

In the CL mode of operation, the Transport service is not expected to provide delivery in transmission order, flow control (control the rate that the packets reach the end system), or error control of the Transport layer PDUs (TPDUs). Hence, the CL Transport protocol is relatively simple.

The functions defined in the OSI Transport Layer, which will serve as the basis for the subsequent comparison study, are shown in the Table 1.

TABLE 1

FUNCTIONS DEFINED IN THE OSI TRANSPORT LAYER [6, 13]

FUNCTION	DESCRIPTION
Assignment to a Network Connection	Links a Transport connection to a Network connection.
Association of Transport Protocol Data Unit (TPDU) with Transport Connection	Associates a TPDU with the Transport connection identifier in order to provide data path to the TPDUs.
Checksum	Procedure that calculates a check value associated with the TPDU. Used to certify user data is not corrupted.
Concatenation and Separation	Concatenates (joins) two or more TPDUs into a single Network Packet Data Unit (NPDU), and separates (reverse of concatenate) a single NPDU into several TPDUs.
Connection Establishment	Initial communication set-up between two computers that occurs before user data can be transmitted.
Connection Refusal	Procedure initiated in response to an inability to perform a connection or to conform with the requirements of a connection.
Error Release	Mechanism used to release connection after an error signal has been received from the Network Service provider.
Expedited Data Transfer	Procedure that guarantees the arrival of an expedited PDU before any TPDUs subsequently transmitted and may bypass data already submitted for transmission.

Explicit Flow Control	Mechanism that regulates the flow of TPDUs to prevent overrunning the responder's capability to receive data.
Frozen References	Procedure to ensure an active connection identifier is not reassigned to a new connection without first closing the old connection.
Inactive Control	Procedure that deals with the unsignalled termination of a Network Connection (NC).
Multiplexing/Demultiplexing	Multiplexing allows more than a single TC to share a single NC.
Normal Release	Procedure to terminate a Transport connection in normal conditions.
Reassignment After Failure	Procedure that links a TC to a new NC in the event that the old NC is lost.
Resequencing	Procedure used to re-sequence TPDUs.
Resynchronization	Procedure executed to recover from a temporary problem in the NC or after a reassignment of a TC.
Retention Until Acknowledgement of TPDU	Mechanism to retain a copy of TPDUs on the sending system until their acknowledgement arrives from the receiving system.
Retransmission on Timeout	Procedure to time a TPDU acknowledgement. It retransmits the TPDU if timeout occurs.
Segmentation and Reassembly	Segmentation is used to divide a large TPDU into several small ones because of network service constraints. Reassembly is the reverse process of segmentation.

Splitting/Recombining	Splitting is accomplished by distributing user data over multiple NCs. This is done to increase throughput (bytes/sec).
TPDU Numbering	Mechanism to number the TPDU in sequence for resequencing and flow control purposes.
TPDU Transfer	Procedure to convey TPDUs from the initiator's system to the responder's system.
Treatment of Protocol Error	Mechanism to handle inappropriate TPDUs.

PROTOCOLS UNDER STUDY

VMTP, XTP and TP4 are protocols that provide the Transport layer functionalities. Figure 4 shows how these protocols fit within the context of the OSIRM.

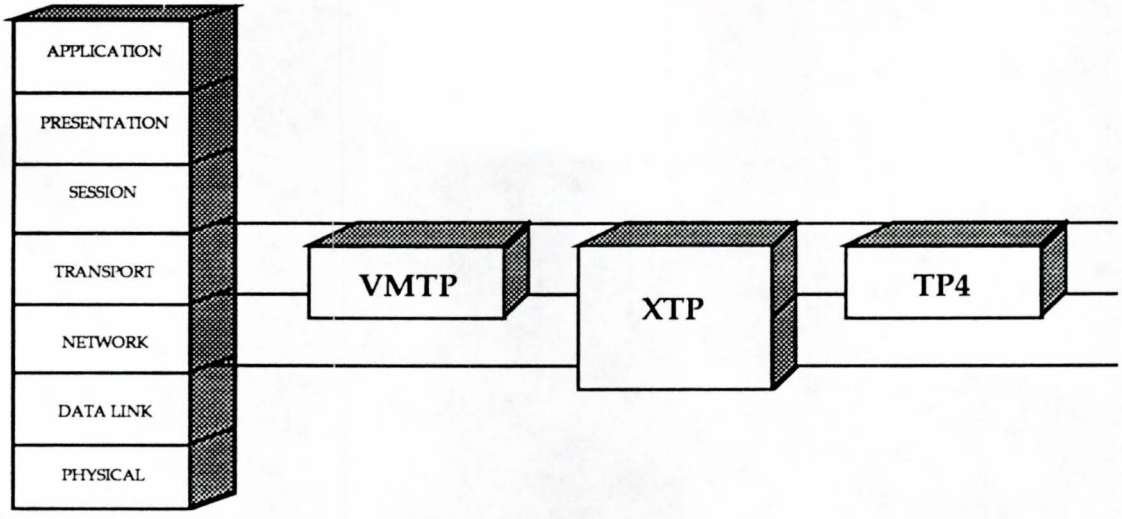

Figure 4. VMTP, XTP and TP4 on OSIRM Context

This section gives an overview of the protocols under study, and lays the foundation for the comparison study in the next section.

Versatile Message Transaction Protocol

The Versatile Message Transaction Protocol (VMTP) is a high performance transport protocol developed by the Distributed Systems Group at Stanford University under the direction of Professor David R. Cheriton. VMTP was developed during research into high performance distributed systems.

With the development and increasing use of distributed systems (multiple processes residing in various sites), the use of communication channels is changing. One example is the distributed, on-line, transaction processing systems (e.g., credit card transactions), which implement a typical "request-response" communication scheme. This example is in sharp contrast with conventional uses of computer network communication which were based primarily on remote terminal access and file transfer (including electronic mail), where the major concerns were the throughput of data and cost of idle connection (for remote terminal access). VMTP was created to more efficiently handle distributed systems and request-response communication.

The VMTP was designed to support Remote Procedure Call (RPC) and other general transaction-oriented communications. RPC is a mechanism through which one host on a network requests another host to execute a process on the remote host, and the remote computer replies to the requesting host with a result from that process. The transaction-oriented communication process is simply a host computer requesting information from another computer, and the latter replying with the requested information.

VMTP was designed to address three categories of deficiencies found in earlier transport protocols in the areas of performance, naming, and functionality [2]. The design of this new generation protocol is based on major technological advancements taking place in computer communications. Today, Local Area Networks (LANs) with multi-megabit data rates, low delay, and low error rates are common. In addition, the future promises even greater high-speed, low-error-rate, wide area connectivity with very high performance switching nodes

and gateways that use low cost memory, processors, and multi-processor technologies. These evolving technologies need a contemporary transport protocol to fully utilize the technology available.

In its operation, VMTP assumes it will be sending messages using an unreliable datagram service. It provides reliable communications to its senders. Figure 5 shows examples of typical VMTP Transactions.

The basic VMTP model provides transactional communications semantics. That is, VMTP provides transport communication between entities on the network via message transactions. A message transaction consists of a request message sent by a client entity to one or more server processes, followed by zero (no response required) or more response messages sent back to the client by the server processes. At most one response message is sent back per server entity. Normally, a client sends a request to a single server and receives a single response, as when a workstation requests a page of a file from a network file server.

Independent of the client/server communication, the VMTP modules also communicate among themselves. Each VMTP implementation has a pseudo-client known as "manager". Figure 6 shows a typical VMTP communications model.

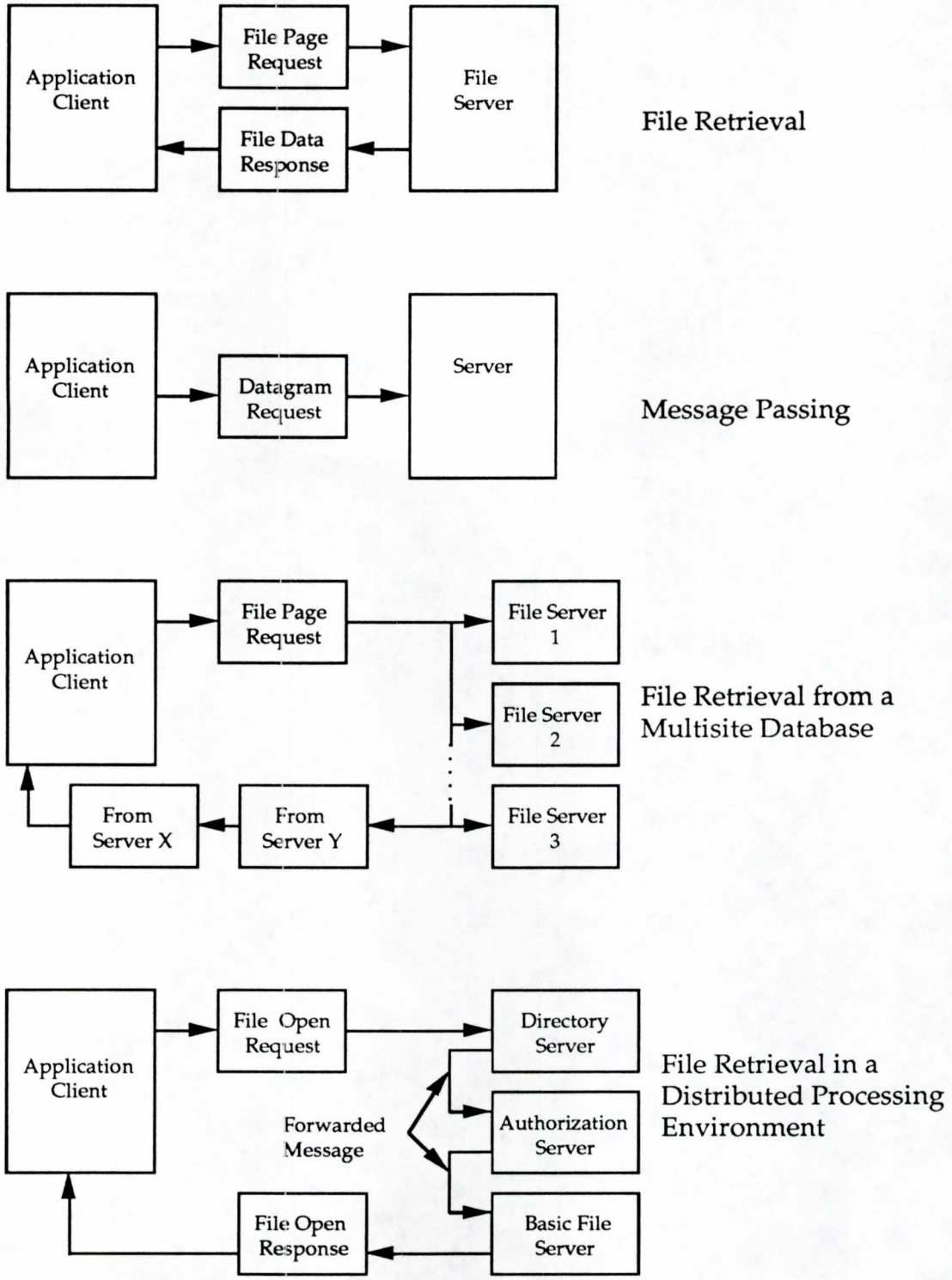

Figure 5. Examples of VMTP Message Transactions

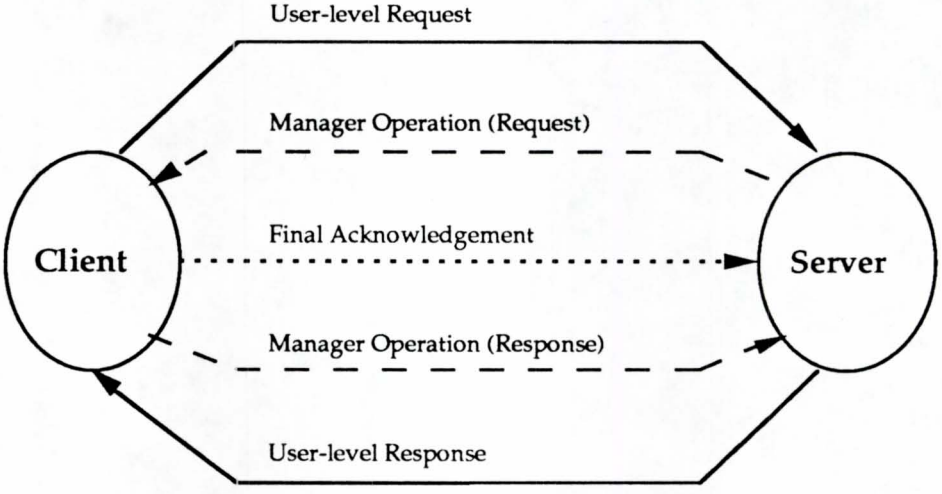

Figure 6. VMTP Communications Model

The VMTP manager is composed of procedures residing in each computer on the network. It is responsible for obtaining status information regarding other entities, and updating message transaction information at the client or the server. Unlike the traditional protocols requiring explicit connection establishment, VMTP sets up connection on demand. The type of connection depends on the type of communication services needed by the application. The VMTP frame format is shown in figure 7.

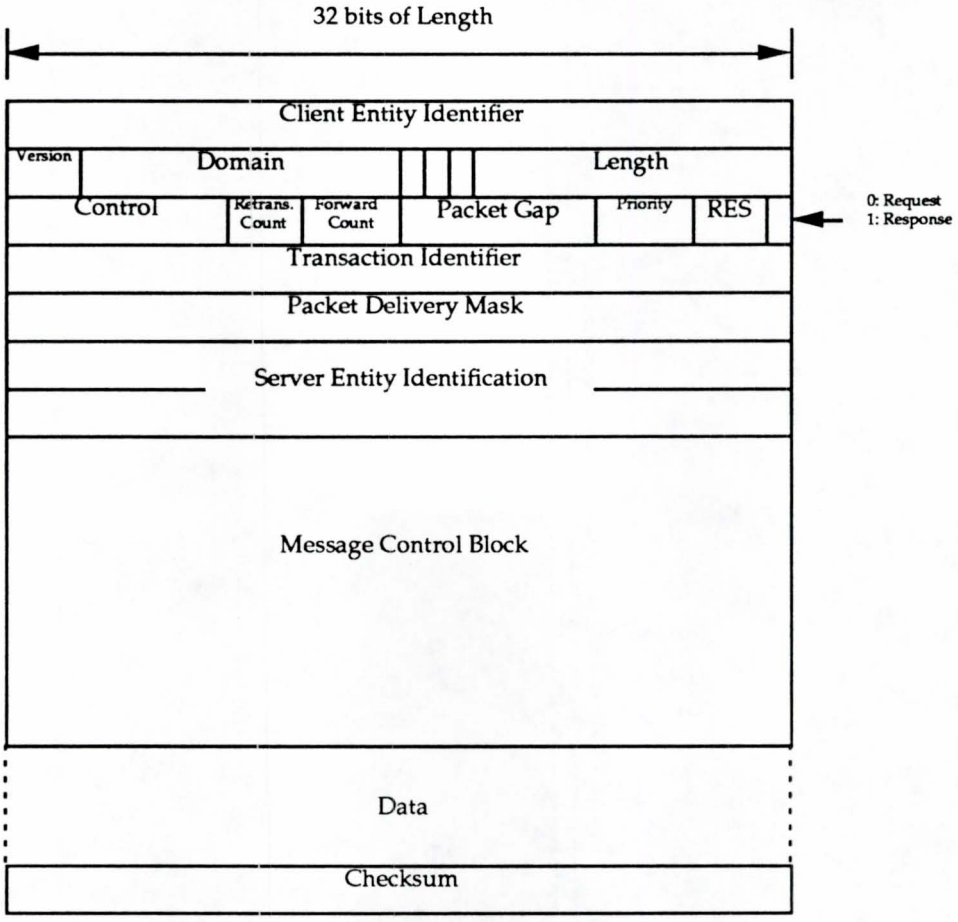

Figure 7. VMTP Frame Format

The VMTP frame is divided into blocks: the Header, the Massage Control Block, the Data Block, and the Checksum. PDU fields of interest in this paper include:

Entity Identifier: a 64 bit field that identifies the Client and the Server for a connection.

Transaction Identifier: a 32 bit field used in conjunction with the Entity Identifier to identify a connection.

Priority: a 4 bit priority identifier used during the communication process to identify message urgency .

Packet Gap: an 8 bit field specifying the interval between two consecutive outgoing packets.

Checksum: a 32 bit error check field holding the value calculated from the checksum algorithm for error checking purposes.

Packet Delivery Mask: a 32 bit field used to acknowledge delivered packets to the sender.

Retransmission Count: a 3 bit field used to maintain the number of retransmissions of a PDU.

Xpress Transfer Protocol

The Xpress Transfer Protocol is an emerging high speed protocol intended for direct implementation in Very Large Scale Integration (VLSI) hardware architectures. It is also known as the Protocol Engine (PE) [4]. XTP was designed by Greg Chesson of Silicon Graphics and refined to its current state by Protocol Engines, Inc., an industry consortium of networking companies, government laboratories, and academic institutions. XTP is targeted at High Speed Local Area Networks (HSLANs).

The XTP is a Transport protocol with unified internetwork services conforming to OSI layers 3 and 4.

The core mechanism in XTP is connection management, controlling two simplex virtual circuits, i.e., controlling two independent unidirectional communication links.

In XTP, a set of state variables contains state information for an instance of XTP at an end-system (host). These state variables are collectively termed the Context. Figure 8 shows an example of an XTP Context. Contexts Ka and Kb (from hosts A and B respectively) are associated. In this association, there is an A to B data stream and a B to A data stream, each used for data packet transmission. Furthermore, there are two control packet streams used for control packets transmission.

There are three primary goals for the XTP/PE system: real-time performance, integrability, and full functionality. Cost reduction is considered a subgoal of integrability, because it allows upgrades on the circuit board without

redesigning the entire circuit. The initial performance goal for the XTP/PE system is a 100 Mbit/s transfer rate between application processes. This is motivated by FDDI technology, since FDDI provides transfer rates in excess of 100 Mbit/s. However, the system architecture has been chosen specifically for the ability to scale up to 1 Gbit/s level. The system is also designed to deliver low latency for real-time applications [5].

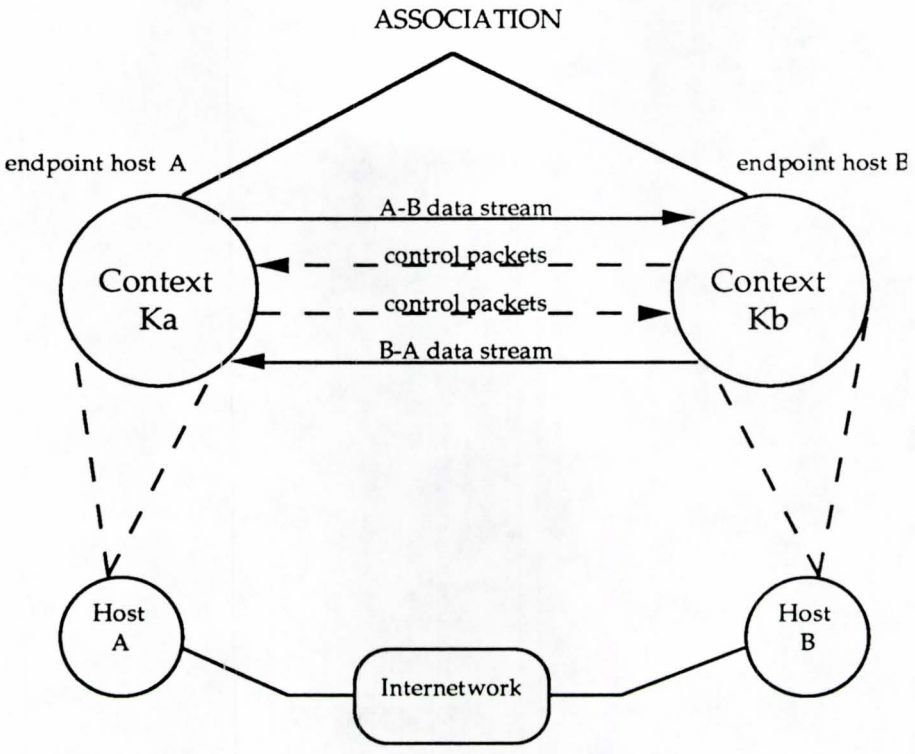

Figure 8. XTP Contexts, Association and Data Streams

The requirements for XTP derive from the communication needs of contemporary and anticipated distributed systems. Distributed systems typically require services such as Remote Procedure Calls (RPC), rapid request/response operations, and transaction-based file access. Such services are naturally provided

by a communications architecture based on connectionless service and reliable real-time, arbitrary-sized, datagrams.

XTP was designed to function as a Transfer protocol, meaning it incorporates services and functions from both transport and network protocols. This study, however, only covers the Transport layer functionalities. The XTP frame is shown in figure 9.

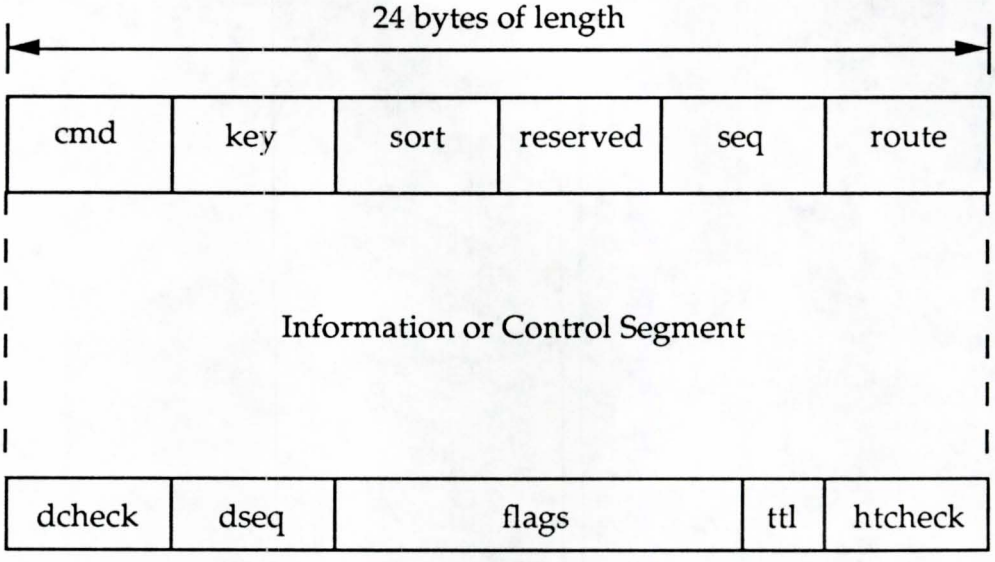

Figure 9. XTP Frame Format

The 24 byte header and 24 byte trailer formats are common to all the XTP packets.

XTP defines 10 packet types: DATA, CNTL, FIRST, PATH, DIAG, MAINT, MGMT, SUPER, ROUTE, RCNTL [12].

Some of the packet types referred to in the next chapter are:

FIRST: packet sent from the initiator to start an association.

DATA: packet containing user data.

DIAGnostic: packet to report error conditions.

ROUTE: packet for exchanging control information.

SUPER: packet formed by conjunction of packets.

Some XTP frame fields referred in next chapter are:

SORT: a 32 bit field used to specify packet priority.

SEQ: a 32 bit field that specifies a sequence number associated with one data stream.

DCHECK: a 32 bit field for a data checksum.

HTCHECK: a 32 bit field for a header/trailer combined checksum.

Transport Protocol Class 4

OSI defines three classes of networks as shown in Table 2.

TABLE 2

NETWORK TYPES IN OSI REFERENCE MODEL

NETWORK TYPES	QUALIFICATION
A	Acceptable residual error rate Acceptable signalling error rate
B	Acceptable residual error rate Unacceptable signalling error rate
C	Unacceptable residual error rate

An error is defined as a missing, distorted or duplicate PDU. A signalled error occurs when the network detects and reports an error. A residual error occurs when an error occurs without network reporting it.

Table 2 indicates that in Type A networks, a negligible number of the PDUs transferred are silently lost and there are very few signalling errors. Type B networks do not silently lose data, but they do have more signalled errors. Type C networks do not provide against data loss, transmission error, or packet reordering.

Within the OSI Transport layer definition there are five classes of transport protocols which are identified by their class number, they are the Transport Protocol Class Zero (TP0), TP1, TP2, TP3, and Transport Protocol Class Four (TP4) [7]. The primary motivation for the five Transport protocols is to

handle the different services provided by Type A, B, and C networks. TP0 is the simplest type of OSI transport protocol. It is intended to be operated over a Type A network, it does not provide error detection or recovery.

TP4 is the most complex of these classes, it is intended for use on a C Type network. TP4 can handle residual error rates unacceptable to T0-T3.

TP4 has been adopted as part of the mandated U.S. Government Open System Interconnection Profile (GOSIP), which is a Federal Government procurement profile for open systems computer network products. This profile is the standard reference for all Federal Government agencies when acquiring and operating data processing systems or services and communication systems or services intended to conform to the OSI protocols. Figure 10 shows how TP4 fits within the U.S. GOSIP.

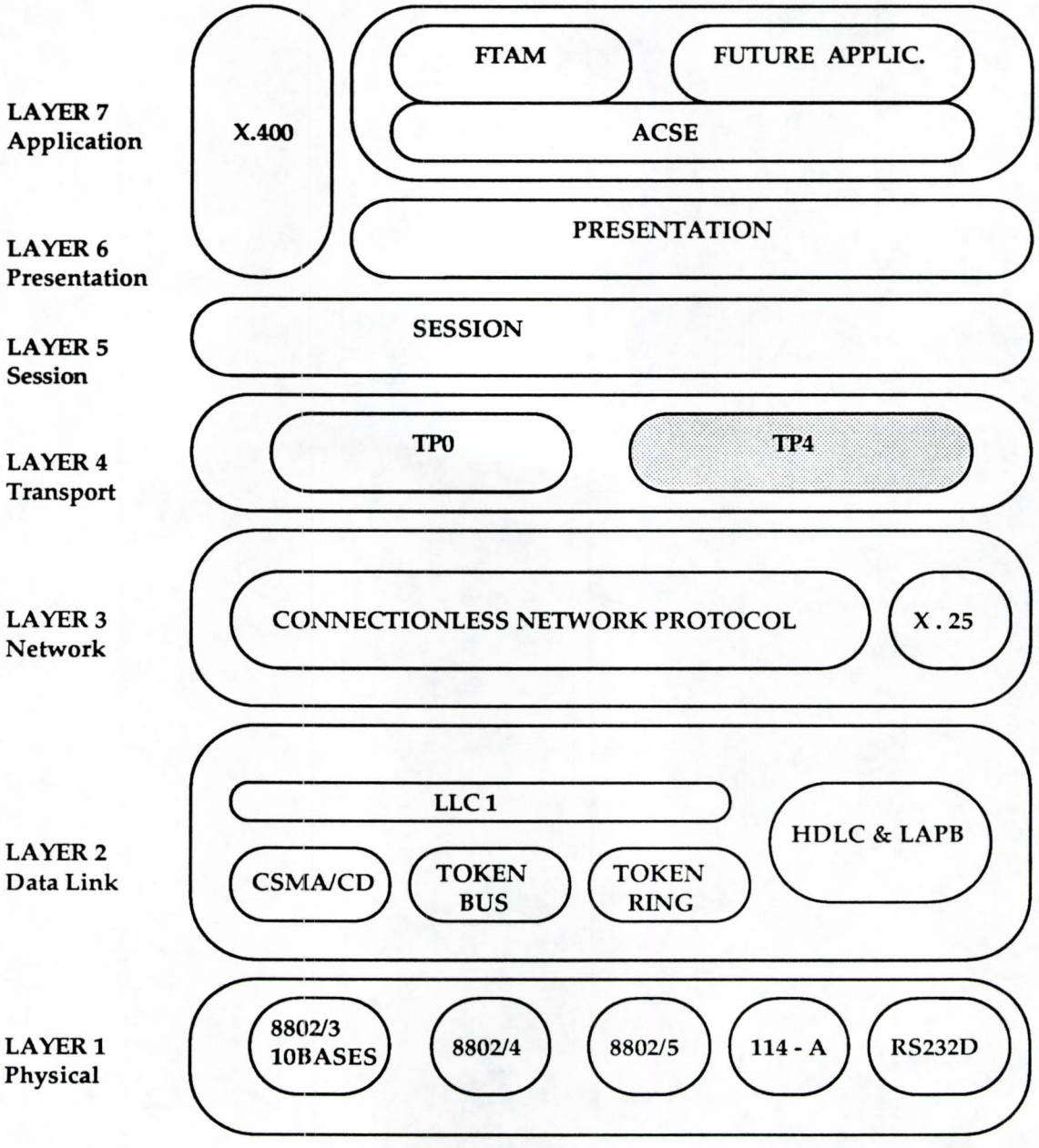

Figure 10. U.S. Government OSI Profile Version 1

CHAPTER 2

APPROACH

VMTP, XTP and TP4 protocols are analyzed using a subset of the OSI Transport layer functions from Table 1. Because of differences in protocol design, not all the functions defined in the OSI Transport layer are applicable to all the protocols under study. Therefore, in order to establish a common base for the comparison, only those functions applicable to all three protocols were used for the comparison.

In the following section, a brief description of each Transport function investigated is given followed by the description of how the three protocols incorporate the function. Finally, the similarities and differences among the protocol specifications, related to that function, are given.

As part of the comparison study, the protocol differences and similarities are identified, as well as advantages and disadvantages of each specification. A summary of the protocol comparison is presented at the end of this chapter.

Association to a Network Connection

The Transport protocol uses services provided by the Network layer. It is through an association to Network connection(s) that the TPDUs are passed to the Network entity(ies).

VMTP

VMTP assumes an underlying network (or internetwork) service providing a datagram transfer. Specifically, VMTP is designed to be implemented on the IP Internet Datagram Protocol, although it may also be implemented as a LAN protocol directly via "raw" network packets. VMTP is assigned the protocol number 81 in Internet IP context.

With a 20 octet IP header and one segment block, a VMTP packet is 600 octets. Thus, a host implementing VMTP implicitly agrees to accept VMTP/IP packets of at least 600 octets .

XTP

XTP suggests many lower layer protocol encapsulations[1] within a Medium Access Control (MAC) frame of the Data Link layer. The XTP packet can directly follow the MAC header, although in many cases additional higher layer protocol headers are required.

To achieve the 8 byte boundary in host memory so 32 and 64 bit computers can access control fields without the overhead of crossing word boundaries, an alignment offset is used to handle various odd MAC header sizes.

---

[1] A technique used by layered protocols in which a layer adds header and trailer information to the upper layer's PDU

The XTP specification defines 5 types of encapsulations [12]:

1- IEEE 802.3 (CSMA/CD) Encapsulation

2- IEEE 802.2 (Logical Link Control) Encapsulation

3- FDDI Encapsulation

4- IEEE 802.5 (Token Bus) Encapsulation

5- IP Encapsulation

TP4

In TP4, it is not until an initial assignment is made to a network connection (NC) that a transport connection (TC) can be established. Assignment is the association of a TC with a NC. TP4 allows the change of the NC in the event that the initial NC is lost [6].

Similarities and Differences

XTP interfaces with a large range of underlying network protocols through encapsulation. VMTP is designed to be implemented on top of the Internet IP. TP4 is usually implemented on top of the OSI IP.

XTP, as a Transfer protocol, can interface directly to the MAC layer, which is not supported by either VMTP or TP4 due to the fact the latter ones are Transport protocols and not suitable for direct interface to the Data Link layer.

Transport Protocol Data Unit Transfer and Acknowledgement

TPDU transfer is characterized by the movement of the user data from a source computer to its ultimate destination. To guarantee reliable transfers, TPDU need to be acknowledged by the receiver.

In a synchronous communication, the sender cannot send an arbitrary number of unacknowledged PDUs (blocking). In a asynchronous communication, the sender does not wait for PDU acknowledgement.

VMTP

A VMTP message is structured with a fixed size header of 64 bytes which includes a Message Control Block (MCB), a data segment of zero or more bytes, and an appended checksum.

Traditional RPC protocol enforces a pure procedural model in that each call blocks until completion. This means communication is synchronized in that one end system waits for its counterpart to finish its process before it proceeds. VMTP provides a mechanism for requesting asynchronous communications service which increases network usage.

VMTP defines four types of request from the initiator [9]:

1 Normal Request - the request is sent, and the client process awaits the server response.

2 Datagram Request - the request is sent, and the client process does not await the server response.

3 Asynchronous Request - the request is sent, and the client process does not await the server response. Instead, the client later requests the response, or responses, from the manager.

4 Multiple Responses Request - the request is sent to a group and the client expects to receive multiple responses.

The minimal message transaction has an empty data segment. In this case, a simple request-response action takes place using only the 64 byte header containing any user data in the Message Control Block. When a data segment is sent, it is transmitted as one or more packets (TPDUs) depending on the data length, with the header and MCB repeated in each packet.

XTP

Of the various XTP packet types, only 2 types are of interest here. The first contains a Control Segment used for managerial information, and the second contains an Information Segment for the user's application data. These packets use the same header and trailer formats .

XTP does not limit the number of bytes in a data packet. Any data packet limit is imposed by lower layer implementation.

Data acknowledgements are driven by request from the transmitter for a receiver control packet, containing the current state of the receiver's connection. The control packet is requested by a field in an outgoing packet header. Since the sender has the knowledge of the underlying network characteristics, it can tune its acknowledgement policy to obtain a minimum overhead. A further

advantage of this transmitter-driven acknowledgement scheme is the control may be transferred from the direct data path to a timer-generated control, reducing the amount of protocol processing required.

TP4

Normal data transfer is accomplished using the Data TPDU (DT TPDU). TP4 does not directly limit the size of TPDUs, but it may not exceed the amount of data the network service provider is capable of conveying between peer entities. For user data exceeding a single TPDU, the user data is partitioned into a sequence of TPDUs. The last TPDU in sequence has an End-of-Transmission (EOT) bit set indicating the end of the data.

TP4 requires acknowledgement of DT TPDUs with acknowledgement (AK) TPDUs. However, several DT TPDUs can be acknowledged with a single AK TPDU.

Similarities and Differences

VMTP and XTP have the capability to transmit with the option of requesting acknowledgements from the receiver. On the other hand, TP4 requires all the PDUs to be acknowledged. The capability to choose the acknowledgement from the receiver is highly desirable for state-of-the-art computer networks, where the integrity of the data transmission over the network is very high, eliminating the need for extra effort to guarantee its delivery. Excess aknowledgement may interfere with real time network services.

Segmentation

Segmentation is the process that divides one PDU into several smaller ones. This is often necessary because of size constrains imposed by the underlying Network services.

VMTP

VMTP messages are broken into packet groups, not exceeding 16 Kbytes. The data segments are viewed as a sequence of data blocks, each block, except the last, 512 bytes. The maximum number of blocks per packet is determined by the underlying network maximum packet size. VMTP requires network support allowing, at least, 512 bytes packet.

XTP

The XTP packets that may be segmented are: FIRST, DATA, DIAG, ROUTE, and SUPER. The remaining packets are small enough such that they do not require segmentation. Once a packet is segmented, it remains segmented until it reaches its destination.

To segment packets, the Information Segment is split into smaller parts, and new headers and trailers are constructed for each smaller packet. The last packet resulting from the segmentation of a larger packet should contain the same alignment padding contained in the original packet. XTP does not specify the segment size for partitioned data. It uses the underlying network limit to constrain the TPDU size.

TP4

User data may be segmented into a collection of Data (DT) TPDUs. The size of the DT TPDU depends on the capability of the network services. Each TSDU is segmented into a DT TPDU which contains a header and a part of the TSDU. The last segment has EOT set indicating the end-of-transmission.

Similarities and Differences

All three protocols define a segmentation function. XTP and TP4 do not specifically define the size of the packet for data segmentation, basing the limits on the underlying network service provider. However, VMTP demands the underlying network provide a packet size of at least 512 bytes.

Connection Establishment

Connection establishment is the set-up done to initiate the communication process. It is in this phase that the network service parameters (e.g., priority, security, flow control) are negotiated.

VMTP

The VMTP protocol is designed to work in a request-response scheme. It uses a minimum of a two packet exchange for short and simple transactions. There are no separate operations for setting up or terminating connections between clients and servers at the Transport layer.

The client sends a service request to a server, the request is processed, and the server responds. For example, a client queries the server for a specific record in its database. The transaction is initiated by sending the request and it is terminated by the server replying to the client with the record requested.

VMTP also provides multi-packet streaming requests and responses for efficient data transfer.

XTP

Context management is the means by which an association context is established in XTP. Before any XTP packet can be sent or received, an active XTP context must exist. Context management defines the state of the XTP contexts and their transition between various states. The context states are: NULL, LISTEN, ACTIVE, and INACTIVE.

Connection setup in XTP uses an implicit two way handshake for each virtual circuit involving the FIRST data packet type and its acknowledgement. The association between contexts is created by transmitting a FIRST packet, which contains the information segment (an address segment and an optional user data segment), from the initiator to the responder.

This causes the responder's context to transition from a NULL state to an ACTIVE state. The initiator has the option to set either SREQ (status request) or DREQ (delivery request) in the trailer segment to request an acknowledgement in response to the initial association. The initiator need not demand an acknowledgement. If either the SREQ or DREQ bits are set, the receiving context responds with a CNTL packet.

The association may persist for multiple message transfers between the contexts. It is up to the initiator to decide how the data transmission should take place.

There are four cases that describe the lifetime of a context:

1 A single, unreliable, message transfer context exists for the duration of the message transmission without regard for correct delivery or acknowledgement.

2 A reliable message transfer context exists long enough to transmit, and possibly retransmit, a message until an acknowledgement indicates correct delivery.

3 A handshake, or transaction, whose context is created to send a single, but arbitrary length, message and is released upon receipt of a single, arbitrary length, reply.

4 A conversation where the context exists for as long as the communication entities wish to remain in contact.

Though the connection established between the contexts is not a duplex connection, but rather two simplex connections with opposite directions, it still provides communications capabilities equivalent to the ones provided by a full duplex connection.

TP4

The connection establishment requires a three way handshake. The Initiator requests a connection establishment by sending a Connection Request (CR) TPDU. The Responder replies with a Connection Confirm (CC) TPDU. A third TPDU from the Initiator is needed to acknowledge the CC. Retransmission may occur if any of these TPDUs are lost or delayed.

Many parameters may be negotiated during the connection establishment phase: Checksum, use of expedited data, explicit flow control, etc.. TP4 establishes a full duplex connection with the possibility of data and acknowledgements flowing simultaneously in both directions.

There is no provision in TP4 for a simple transaction model (Datagram) communication. The initiator is required to both initiate and close the connection every time it needs to perform a transaction.

Similarities and Differences

TP4 and XTP define connection establishment and connection release at the Transport level, whereas VMTP leaves this function to the Application processes. VMTP and XTP are able to provide a fast message transaction between end systems or even datagram transmission whereas TP4 requires an explicit three way handshake to establish a connection. For bulk data transfer, the three protocols are analogous. For RPC applications, XTP and VMTP are superior in that they allow faster connection establishment and release.

For connection oriented protocols such as XTP and TP4, an initial exchange of state information precedes the actual communication. Typically, the connectionless protocol packets should contain sufficient information to provide the Connection State. However, the VMTP leaves this managerial work to the VMTP protocol manager implemented in each host. This allows the protocol to discard the state when the communication is idle, since it can obtain the state information any time it is needed.

Connection Release

Connection Release is the process that takes place at the end of a communication process. It is required to release the computing resources allocated to a specific communication process and making them available to other processes.

A normal connection release can be Graceful or Abrupt. A graceful release is agreed to by the Initiator and the Responder, but an abrupt release is a unilateral release which forces the counterpart to close its connection.

VMTP

VMTP does not have an explicit connection establishment phase. Therefore, an explicit connection release is not needed. All connection release work is performed at the application processes level.

XTP

Connection release in XTP is coordinated using three flags: RCLOSE, WCLOSE, and END. During the normal release, the sender of the data stream indicates the connection release by setting the WCLOSE bit on the last packet destined for the receiving host. The receiver responds with RCLOSE when all the data has been received. It is possible for the receiver to request retransmission of missing packets before an RCLOSE is sent. Note that in two simplex connections between nodes, data would be transmitted in both directions and each direction must be closed independently.

The END flag is set in an outgoing packet to signal the remote host that the local host released and closed its end of the connection. Therefore, END is set in the final packet transmitted and indicates the context has been terminated and no further packets can be exchanged.

TP4

TP4 defines primitives applied to the connection release, **T-Disconnect.Request** and **T-Disconnect.Indication**. The TP4 Transport connection release dissolves the connection immediately. It is a connection abort which does not negotiate the connection release.

In TP4 the initiator requests a connection release by sending **T-Disconnect.Request**. The Disconnect.Request arrives at the responder as a Disconnect.Indication and the responder is forced to close the connection without a chance to negotiate the connection release. However, TP4 guarantees that data will be delivered to the transport provider at the destination. This does not necessarily mean the transport service user will receive the data. The data may be buffered by the destination transport provider awaiting receipt by the destination user.

Similarities and Differences

XTP defines an "agreed release" by both parties of the connection (graceful release), TP4 defines a "non-negotiated" release by the initiator of the connection (abrupt release), and VMTP does not need an explicit connection release at the transport level.

Association of TPDU to Transport Connection and TPDU Numbering

The TPDUs are associated with Transport connections through connection identifiers. The association allows the TPDUs to be transmitted to a destination in a consistent manner (i.e., it provides an unambiguous identification to the TPDU, differentiating TPDUs directed to different destinations). TPDU numbering is required to provide a means of reordering the TPDUs should they arrive out of the transmission order.

VMTP

In VMTP each message transaction is uniquely identified by a transaction identifier (Client,Transaction) pair. The 32-bit transaction identifier is initialized to a random value when the client entity (VMTP communication endpoints) is allocated its entity identifier. The transaction identifier is incremented at the end of each message transaction. All responses with the same specified (Client, Transaction) pair are associated with this request.

Each packet group is identified by a transaction identifier. Normally, a message is sent as a single packet group. That is, its data segment size is less than 16 Kbytes. For larger data segments the offset of a packet group within the message is indicated by its transaction identifier relative to the transaction identifier of the first packet group in the message.

Response messages are paired to the request message by specifying the same client identifier and transaction identifier(s) as specified in the request. Thus, there is a single sequence number used for bi-directional communication.

XTP

An Association is the name given to a pair of active contexts and the data streams between them. When an initial (FIRST) XTP packet has been both sent and received, two contexts are created (one at each endpoint). The contexts know about each other and share an interpretation of <MAC address, key, route> read from the XTP packets.

XTP defines a data stream as an arbitrary length of sequenced bytes represented by a 32 bit sequence number. The initiating host of an association assigns two arbitrary starting-point values: one for each data stream (incoming and outgoing). The initiating host uses its starting-point value in its initial FIRST packet and communicates the value it has assigned to the other host's outgoing data stream (the SEQ value).

Two classes of information are counted in the host's outgoing data stream: higher-layer application data and address information included within the FIRST packet of a association. Two types of XTP packets contain information that consumes sequence space as they are sent: FIRST and DATA packets. None of the information of other types of XTP packets consumes sequence space.

TP4

TP4 associates TPDU with the transport connection through the use of a (source address, destination address) pair. The referred pair is carried by the TPDUs for specific connection in their headers and it identifies a unique connection between two transport peer entities.

In TP4, an initial 32 bit sequence number is negotiated during the connection establishment. Subsequently, all TPDUs transmitted are numbered in sequence starting with an initially agreed value.

Similarities and Differences

All three protocols provide a 32 bit packet numbering to track the packets for packet error recognition and retransmission. VMTP and TP4 provide a numbering scheme for entire packets transmitted, whereas XTP numbers by bytes consumed in the FIRST and DATA packet Types.

Expedited Data Transfer (Prioritization)

Prioritization is required to provide a means to coordinate the processing of various TPDUs taking into account their relative urgency.

VMTP

VMTP provides a 4 bit priority identifier for processing each type of request for both transmission and reception. The interpretation is:

1100	urgent/emergency
1000	important
0000	normal
0100	background

Viewing the high-order bit as a sign bit (with 1 meaning negative), low values are high priority and high values are low priority. The lower two bits indicate additional (lower) gradations for each level creating 4 grades of priority in each level.

XTP

XTP leaves implementation of the sort/priority functions as optional. The value contained in the sort field is only interpreted when the SORT bit is set in the header.

The SORT field in an unsigned 32 bit field specifying ordering. All zeros is the highest priority, followed by increasing integers, each representing a lower

value of priority. Thus, XTP supports 2^{32} levels of priorities with all ones being the lowest priority.

Normally, the outgoing packets follow the sequence in which they were processed by the higher-level application in a first-in-first-out basis (FIFO), and all the packets carry the same priority. Upon reception of packets of higher priority, these are processed first. Only after all higher priority packets have been completed will the server return to lower priority packets.

In XTP two preemptive schedulers are defined, one for incoming packets and one for outgoing packets.

TP4

The expedited data TSDU transfer is only available when requested and agreed to during the connection establishment. When available, this service is bi-directional.

The T-Expedited.Data delivers the Transport Service (TS) users data through an out-of-band channel. This data is delivered before any subsequently submitted normal TSDU on that transport connection. It does not guarantee its delivery before normal data already queued.

Similarities and Differences

VMTP and XTP define the priority as a field within the PDU header of the normal data, while TP4 defines a special PDU format for expedited data.

XTP provides 2^{32} levels of priority. VMTP provides 16 levels of priority, and TP4 provides two levels of priority.

XTP and VMTP are more suitable for applications requiring several levels of communication priority processing because of their richness in PDU priority specifications. TP4 lacks communication priority processing, allowing only the transmission of break signals for unusual situations.

Explicit Flow Control

Flow control is required to limit the rate in which the TPDUs are transmitted. It is through flow control that two computers with different processing speeds can communicate without flooding.

There are several flow control mechanisms available, including:

Sliding Window : The receiver specifies the number of packets it is willing and able to receive and this information it conveyed to the sender in the form of credits.

Packet Gap : The receiver specifies the interval between two consecutive packet transmissions.

VMTP

VMTP uses rate-based flow control (i.e., controls the speed of packet transmission). This is achieved by spacing the packet group with inter-packet gaps, thereby reducing the arrival rate. The rate control can be implicit or automatic.

Implicit rate control takes place whenever a client or server explicitly communicates its desired inter-packet gap time. In this case it is assumed the two end nodes have knowledge of the counterpart packet generation capability, and also have some knowledge of the type of gateways on the network.

The automatic rate control process estimates and adjusts inter-packet gap time so as not to overrun a server or intermediate nodes. For example, if the

receiver requests retransmission of every fifth packet, the sender assumes an overrun is taking place and packets are being dropped. The sender can then reasonably increase the inter-packet gap time to deal with that situation. However, the sender can also periodically attempt to decrease the inter-packet gap time whenever no packet loss occurs by pushing the flow rate as high as possible without loosing packets. In this way, the maximum packet rate the receiver or the intermediate gateways can accommodate would be reached.

The inter-packet gap control information is an 8 bit field in the packet header. The gap is expressed in 1/32nd's of the Maximum Transmission Unit (MTU) packet transmission time. Therefore, the minimum inter-packet gap is zero and the maximum gap that the field can represent is slightly less than 8 packet times.

XTP

The volume of XTP data output is regulated by an end-to-end sliding window mechanism. The rate or speed at which XTP sends packets into the local network, as well as the rate at which an intermediate node forwards packets, is regulated by an independent, timer-based mechanism.

In XTP, the receiver's flow control parameters are included in the control packets sent from the receiver to the sender. The parameters include the ALLOC sequence number that constrains the highest sequence number the receiver will accept. Other parameters such as DSEQ (delivered sequence number), RSEQ, (received sequence number) also work together to control the data flow.

If the receiver and the sender have inherently different processing speeds (e.g., different XTP implementations or different hardwares), a rate type of control is necessary to prevent the receiver overrun. The rate control mechanism restricts the size and time spacing of bursts of data from the sender. Within an interval, the number of bytes the sender transmits must not exceed the ability of the receiver (or intermediate routers) to decipher and queue the data by creating gaps in the receiver data stream. The parameters RATE and BURST dictate the maximum number of bytes the receiver will accept in a one second time interval, and the maximum number of bytes the receiver will accept per packet burst, respectively.

TP4

TP4 uses a sliding window for flow control. The initial credit parameter is set during the connection establishment phase. This window is subsequently manipulated by the receiver in order to control the flow from its peer. In general terms, the window imposes a bound on the number of DT TPDUs that can be transmitted without explicit acknowledgement. The receiver transmits the upper and lower edges of the window indicating the TPDUs received and the highest numbered TPDU expected, respectively.

Similarities and Differences

XTP provides a byte-oriented flow control while VMTP and TP4 provide a packet-oriented flow control.

TP4 provides a sliding window flow control. VMTP provides a packet gap flow control. XTP provides both.

XTP provides the best flow control allowing the receiver to specify, at the byte level, an acceptable data flow.

Checksum

A checksum provides a means to validate received TPDUs. The checksum algorithm is calculated at the sending and receiving sites. The receiver compares its computed checksum against the value contained in the CHECKSUM field of the TPDUs.

A PDU checksum is initially calculated by the sender and placed in the outgoing PDU. The receiver applies the same algorithm to the entire PDU, not including the checksum, and should get a match if there are no errors.

VMTP

The checksum field in VMTP is 32 bits and is appended to each packet, immediately after the data field. The checksum consists of two 16 bit ones-complement sums, covering different parts of the packet.

The checksum normally covers both the header and the segment data. Optionally, the checksum may apply only to the packet header.

A zero checksum field indicates no checksum was transmitted with the packet. VMTP allows the checksum to be turned off when there is a host-to-host error detection mechanism and the VMTP security facilities are not necessary. One example of such case is running VMTP over an Ethernet network, which provides sufficient reliability to the Transport interface.

XTP

XTP uses two XTP packet checksums. The DCHECK is a 4 byte checksum for the packet data field. The HTCHECK is a 4 byte header and trailer checksum. The checksums are placed in the last few bytes of the XTP packet trailer so the checksum calculation can be concurrent with packet transmission or reception. When either checksum indicates a packet error, the receiver discards it. If the source was known, the receiver could immediately inform the sender process the packet was corrupted in transit, allowing the sender to begin retransmission. Setting the NODCHECK or NOCHECK indicates the checksum is disabled in the current packet for data or header/trailer respectively.

TP4

TP4 mandates a checksum for all CR TPDU and for all TPDUs when the checksum option is selected during the connection establishment. The TP4 checksum mechanism employs a 16-bit checksum included as a parameter in each internet PDU.

Similarities and Differences

XTP and VMTP place the checksum at the end of the packet while TP4 places the checksum in the header of the packet.

All three protocols allow packet checksum, but TP4 mandates the checksum calculation for the CR TPDU.

The checksum mechanism used by TP4 is too complex [11] for the reliability network services can provide today.

The checksum in the TP4 packet header is less efficient than those placed in the trailer segment in the VMTP and XTP packets, because XTP and VMTP can transmit the PDU while computing the checksum, TP4 can only transmit the PDU after computing the checksum.

Retransmission on Timeout

Retransmission may be required to recover from packet loss at any time during the communication process. The retransmission criteria can be time-based or request-based. Time based retransmission is realized by setting a timer on packet transmission and either clearing the timer on packet acknowledgement or retransmitting the packet on timer expiration. The request-based retransmission is realized through a retransmission request sent from the receiver.

Retransmission can take place in two manners: *selective* and *go-back-n*. In selective retransmission, only lost packets are retransmitted. The go-back-n mechanism requires all packets to be retransmitted starting from the first lost packet detected.

VMTP

VMTP provides selective retransmission to recover from packet loss from overruns or network failure. A message, whether Request or Response, is sent as one or more packet groups. A packet group is one or more packets, each containing the same transaction identification and message control block. The data segment of each packet group is represented by a 32 bit mask contained in the packet header. The receiver of a packet group maintains a cumulative delivery mask of blocks received for a particular transaction. On timeout with an incomplete packet group, an acknowledgement is sent indicating the blocks received and explicitly triggering retransmission of the missing blocks. In this fashion, the receiver can request retransmission of any selected portion of the packet group, avoiding the retransmission of the entire packet group.

VMTP recommends a maximum of 5 retransmissions before giving up [3].

XTP

XTP defines a wait timer (WTIMER) to compute the time an XTP sender will wait for a response to an SREQ or DREQ before retransmitting SREQ.

XTP defines both go-back-n and selective retransmission. The go-back-n is used when the receiver forces the sender to retransmit both lost data and received data. The selective retransmission method is used when the receiver keeps track of all the lost data (up to 16 separate gaps) for any given context. This method allows the sender to transmit a minimum number of control (CNTL) packets.

TP4

TP4 defines a timer (T1) to run for each DT TPDU transmitted. The value of T1 is based on the Round Trip Time (RTT) plus the remote acknowledgement time plus the local TPDU processing time. T1 is set on transmission, and retransmission takes place if T1 expires before the acknowledgement is received. The number of retransmissions is limited, after which the sender invokes the release procedure and informs the TS user of the failure.

TP4 uses the go-back-n retransmission method in which the sender retransmits the sequence of packets from the first packet not acknowledged. Selective retransmission is not defined in TP4.

TP4 does not provide the capability for the receiver to request a retransmission.

Similarities and Differences

VMTP uses selective retransmission, TP4 uses go-back-n retransmission, and XTP provides both.

XTP provides more control on the data flow than provided by the other two protocols. However, in a more reliable network connecting similar computers, the difference would be negligible given the low data loss and homogeneity of the computers participating in the communication.

Summary of the Protocol Comparison

TABLE 3

COMPARISON CHART

CHARAC-TERISTIC	VMTP	XTP	TP4
Connectivity	CL	CO/CL	CO
Net. Interface	Internet IP	MAC Layer Encapsulation	OSI IP
Reliability	Reliable/ Unreliable	Reliable/ Unreliable	Reliable
Transaction	Yes	Yes	No
Segmentation	Yes	Yes	Yes
Connection Process	Not Applicable	Implicit Handshake	Explicit Handshake
Release Process	Not Applicable	Graceful	Abrupt
Numbering	32 bit Packetwise	32 bit Bytewise	32 bit Packetwise
Priority	2^{32} Levels	16 Levels	2 Levels
Flow Control	Packet Oriented - Packet Gap	Byte Oriented - Packet Gap and Window	Packet Oriented - Window
Checksum	Optional - Placed in the Trailer	Optional - Placed in the Trailer	Optional with exception of CR TPDU - Placed in the Header
Retrans-mission	Selective	Go-Back-N and Selective	Go-Back-N
Types of PDUs	2	10	10

CHAPTER 3

FUTURE TRENDS AND THE EVOLUTION OF VMTP, XTP AND TP4

The protocols compared in this study are relatively new, especially VMTP and XTP. They are changing to incorporate functions to support real-time applications, as well as new types of applications needing special Transport layer services.

VMTP is changing some of the features defined in the original design, and is adding new features. Some of the future additions will include:

- Timestamps to better control the packets traffic.
- Division of transmission and control information to enable a more efficient data transmission.
- No fixed data block size, to provide more flexibility to the individual packet transmissions.

XTP has changed a great deal since its first release. It is progressing toward greater integration levels and higher performance. The number of VLSI packets needed in the Protocol Engine subsystem will be reduced with time.

XTP is being redefined by ANSI working group X3S3.3 (High Speed Protocol project) as a future standard. The process is in its initial phase but looks very promising.

Several major vendors are already implementing TP4 (e.g., Kodak, IBM) even though it is still in flux. Its promising specification includes features of great interest to many corporations (e.g., network management, network security).

TP4 is still lacking Quality of Service assurance capabilities. Further, there are many issues such as network management and security to be defined within the context of internetworking. Finally, TP4 has to evolve to meet the network requirements of today's applications such as real-time, geographically disperse defense simulation internetworking .

CHAPTER 4

CONCLUSIONS

State-of-the-art Transport protocol designs, such as VMTP and XTP, rely on today's communication networks being more reliable (low data loss rate) compared to older networks (e.g., ARPANET). The design of these protocols are more concerned with performance issues than with data loss. This is manifested by using simpler Checksum algorithms for error checking and placing the Checksum field in the Trailer of the PDUs.

VMTP and XTP are the new generation protocols that promise to eliminate some of the problems that earlier protocols, such as TP4, cannot solve. VMTP and XTP simplify the request/response process by means of optimizing the communication process. This is achieved by minimizing packet exchange between communicating devices. VMTP and XTP can transmit with the option of requesting an acknowledgement from the receiver. In the case of XTP, a packet acknowledgement can also be implicitly realized on the next packet arrival.

The design of VMTP and XTP takes into account the fast pace of changes taking place in the Application layer requirements; e.g., the capability of the XTP protocol to be implemented in hardware to improve its performance over a software implementation. Another example is the fact that VMTP specifies only two packet formats, one for the sender and one for the receiver, which allows easier upgrades on the existing VMTP implementations.

Although VMTP and XTP promise to solve many problems, their effectiveness cannot be gauged until they are implemented. Only practical experience will show if the new approaches can actually supplant the earlier generation of transport protocols.

GLOSSARY OF TERMS [1][8]

ARPANET: A packet switched network developed in the late 60's and early 70's. The "grandfather" of today's Internet. ARPANET was decommissioned in June 1990.

Broadcast: A packet delivery system where a copy of a given packet is given to all hosts attached to the network.

Checksum: The sum of a group of data provided with the group for error detection purposes.

Distributed Systems:

A system in which the system intelligence is disbursed over several remote sites.

Duplex: A facility which permits transmission in both directions simultaneously (sometimes referred to as full duplex, contrasted with half-duplex).

Electronic Mail:

Delivery of mail, all or part, by electronic means.

Encapsulation:

The technique used by layered protocols in which a layer envelopes higher level PDUs by adding header and trailer information.

End System: An OSI system which contains Application processes capable of communicating through the OSI protocols.

Entity: OSI terminology for a layer protocol machine. An entity within a layer performs the functions of the layer within a single computer system, accessing the layer entity below and providing services to the layer entity above at local service access points.

Header: Information appended to the front of a block of user data as part of the protocol envelope.

Internet: A collection of networks connected by a set of routers which allow them to function as a single virtual network.

LAN: Local Area Network. A configuration of transmission facility to provide communications within a limited geographical area.

File Server: In a Local Area Network, a file server is a station dedicated to providing file and data storage to other stations on the network.

NSDU: Network Service Data Unit.

Overhead: All information, such as control, routing, and error checking that is sent over a network in addition to user transmitted data.

Overrun: Loss of data because a receiving device is unable to accept data at the rate it is transmitted.

Packet: Group of bits treated as a unit for communication routing purpose.

PDU: Protocol Data Unit. This is OSI terminology for a data communication packet. A PDU is a data object exchanged by protocol machines (entities) within a given layer. PDUs consist of both Protocol Control Information (PCI) and User Data.

Protocol: A formal description of messages to be exchanged and rules to be followed for two or more computer systems to exchange information.

Recovery: The necessary actions required to bring a system to a predefined level of operation after a failure.

Router: A system responsible for making decisions about which of several paths network (or Internet) traffic will follow. To do this it uses a

routing protocol to gain information about the network, and algorithms to choose the best route based on routing metrics.

RPC: Remote Procedure Call. A popular paradigm for implementing the client-server model of distributed computing. A request is sent to a remote system to execute a designated procedure, using arguments supplied, and the result is returned to the client.

Simplex: A communication system or equipment capable of transmission in one direction only. Permits transmission in only one direction from A to B but not from B to A.

Three-way-handshake:

A process whereby two protocol entities synchronize during a connection establishment through connection request, request acknowledgement and acknowledgement confirm.

Throughput: The total useful information processed or communicated per time unit.

TPDU: Transport Protocol Data Unit.

Trailer: Control information appended to the end of a block of user data as part of the protocol envelope.

TSDU: Transport Service Data Unit.

VLSI: Very Large Scale Integration.

BIBLIOGRAPHY

[1] American Institute, Data Communication Terms (Institute for International Research, 1989)

[2] Cheriton, David R. and C. A. William, VMTP as the Transport Layer for High-Performance Distributed Systems (IEEE Communications Magazine, June 1989)

[3] Cheriton, David R., VMTP: Versatile Message Transaction Protocol Specification (Request for Comments 1045, 1988)

[4] Chesson, G., Protocol Engine Project (UNIX Review, Sept/1987)

[5] Chesson, G., XTP/PE Overview (IEEE Proceedings, 1988)

[6] Henshall J. and S. Shaw , OSI Explained: End-to-end Computer Communication Standards (Chichester: Ellis Horwood Limited, 1988)

[7] International Organization for Standardization, Open Systems Interconnection - Basic Reference Model (ISO 7498, 1984)

[8] Jacobsen, O., A Glossary of Networking Terms (Request for Comments 1208, 1991)

[9] Mason, W. A., VMTP: A High Performance Transport Protocol (California: ConneXions, June 1990)

[10] Ministre de la Defense, Republique Francaise, Military Real Time Local Area Network: GAM-T-103 (France, 1987)

[11] Piscitello, David M., OSI Middle Layers: Network and Transport (Interop, Inc., 1990)

[12] Protocol Engines, Inc., XTP Protocol Definition Rev. 3.5 (California: Protocol Engines, Inc., 1990)

[13] Stallings, W., Computer Communications: Architecture, Protocols and Standards (IEEE Computer Society Press, 1987)

[14] Stallings, W., Handbook of Computer Communications Standards Vol. 1 (Indiana: Howard W. Sams & Company, 1989)