DESCRIPTION OF A PLANNED FEDERAL INFORMATION PROCESSING STANDARD FOR TRANSPORT PROTOCOL

John F. Heafner
Robert P. Blanc

National Bureau of Standards
Washington, D. C. 20234

ABSTRACT

The National Bureau of Standards has developed service and design specifications for transport and session protocols for use in computer system and network procurements. These protocols reside in layers four and five of the International Organization for Standardization's (ISO) Reference Model for Open Systems Interconnection. This paper describes the services, interfaces, and internal behavior of the transport protocol. The transport (and session) protocol specifications were derived from the most recent developments within ISO on these protocols. Specific features were selected based on the needs of the agencies of the Federal Government within the United States, but they are consistent with the needs of any large organization engaged in the procurement or development of networks of heterogeneous computer systems.

TRANSPORT AND THE NBS NETWORKING PROGRAM

The Systems and Network Architecture Division of the National Bureau of Standards (NBS) has ongoing computer networking programs [ICCC 80] to develop Federal Information Processing Standards in three major areas: computer network protocols, computer-based office systems, and local area networks. The goal of these programs is to provide a structure which supports distributed processing within the Federal Government, where system components can be procured competitively without the constraint imposed by incompatibilities.

A primary objective is to specify standards which will result in the availability of off-the-shelf implementations. It is therefore necessary to define standards based on international developments so that implementations can be marketed internationally. It is also necessary to issue the standards on a timely basis that will result in minimal retrofitting of existing networks.

The network protocol program is developing protocol standards for internetworking, transport, session control, file transfer, virtual terminal, and remote job entry. The transport protocol, whose revised design has recently been completed, is the subject of this paper.

The transport protocol design [BURJ 81] and an implementation were developed by Bolt Beranek and Newman, Inc. (BBN) under contract to NBS. The technical work performed by BBN was done in conjunction with NBS and with the voluntary standards organizations ANSI, ISO, and CCITT. We shall describe the transport's services and internal workings in some detail. However, the formal specification cited above should be consulted for a complete and accurate description.

The protocol features are based on the requirements of the Federal Government. For example, the features of the extended class transport (described below) are actually a superset of the Department of Defense's TCP-4 [TCP 80]. Thus, the draft standard provides the functionality of TCP-4. The protocol internals, such as addressing, sequencing, and encoding, are based on the latest developments within ISO. The specification, however, includes design details not found in the ISO documents in order that the specification may be referenced by users in procurements and may be implemented by manufacturers.

INTRODUCTION TO THE TRANSPORT PROTOCOL

Architectural Model

In the development of complex standards such as transport it is useful to follow a guideline which abstractly models the protocol. Such a guideline exists for

transport in the form of the ISO Reference Model for Open Systems Interconnection [ISO 80]. This model partitions the computer network functions and services into seven protocol layers which collectively address network protocol modeling. Each host environment (e.g., mainframe and front-end processor) on a network supports the seven layers such that peer layers in different systems may communicate with each other across the network. One end of the model deals with physical, electrical, and mechanical interfaces while the other end concerns services provided to applications. Each succeeding layer builds upon the services provided to it by the layer below and upwardly offers an enhanced service. Transport resides in the middle of the architecture, at layer 4.

Purpose of a Transport Protocol

Transport provides the end-to-end data transport service where the ends can be thought of as residing in the host installation environment, or host nodes of a network. Certainly, details differ depending upon user requirements and network support, but typically a transport might provide both a bulk data and a transaction data transfer service between two users that may be located on different host computers on different networks. Transport would also guarantee integrity of the user's data.

About a dozen other requirements, emanating from Federal Government organizations' data transport needs, were placed on the development of this proposed standard transport. These can be seen throughout the remainder of this paper.

Classes of Services

The proposed transport offers two classes of service, basic and extended. The main difference between them is that, over the same network, extended class provides higher reliability (through additional functionality) than does basic class. The need for two classes of service derives from the nature of the underlying network, the application requirements, and the host terminal equipment.

The relationship between service class and communications network can be explained in terms of the concept of service quality. It is desirable to ensure reliable in-sequence delivery of data between transport users (t-users) by guarding against data loss, duplication, corruption, and misorder. These reliability assurances (service quality) can by specified at different layers within the architecture. For example, a reliable transport service can be obtained by using a network containing much of the error checking mechanisms, coupled to a transport that uses these mechanisms without duplicating them. Conversely, the error controls can be located in the transport instead of in the network. Since both reliable and unreliable networks are in use, this difference leads to two classes of transport to insure uniform service quality.

Many network users require end-to-end integrity for all data transmission. By definition of end-to-end, as host installation to host installation, this integrity is obtainable only at the transport layer. Such users will benefit from a transport containing the reliability assurance mechanisms that operate over a simple datagram network. This approach avoids duplication of function. Other network users have reliability requirements that can be met by X.25-type virtual circuit networks. Again, to avoid duplication of function, these users will benefit from a much simpler transport. Fundamentally, this is the difference between the basic and the extended classes of transport protocol. In addition to performing all the functions of the basic class, the extended class reorders data units arriving out-of-sequence, detects duplicates, and discovers damaged data. Figure 1 depicts this concept of uniform service quality based upon varying network characteristics.

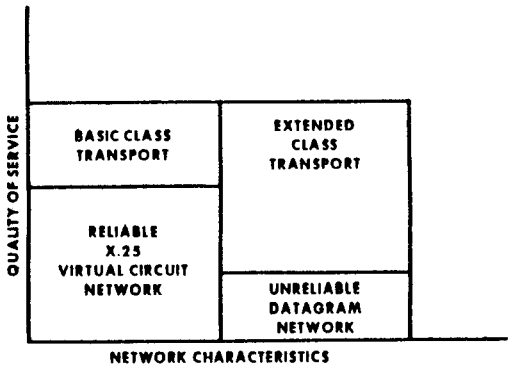

Fig.1 Service quality based on network characteristics

Local networks are also of interest. Due to the inherent reliability of some of the local network technologies, it might be efficient to use the basic transport with a datagram network. However, when such networks are concatenated and interconnected with larger networks and public data networks, then the extended transport is essential for end-to-end reliability.

A relationship between the application and transport service class stems from the terminal equipment being used. Specifically, the basic class, due to its relative simplicity, is more readily adaptable to efficient LSI implementation in displays or display clusters. Figure 2 shows a possible implementation of the basic class in a terminal concentrator where the terminal user employs the basic class on the local network and the extended class over concatenated networks.

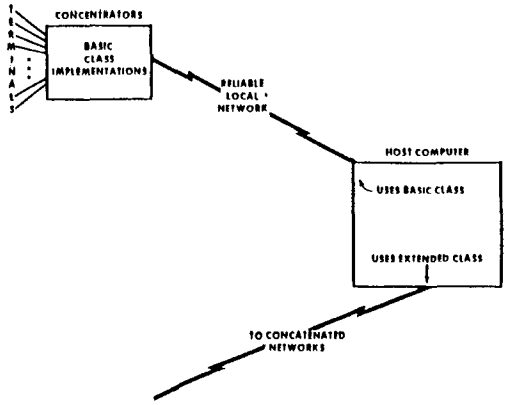

Fig. 2 Terminal concentrator use of basic class

The proposed Federal standard, then, specifies two service classes, basic and extended. The present work of ISO describes five classes, 0-4, where class 0 is the most primitive, providing only connection establishment and data transmission, and class 4 is the most capable, providing multiplexing, flow control, and extensive error control. The Federal basic class corresponds to the ISO class 2 and the extended class corresponds to class 4.

Consistent with the ISO class structure for transport, the NBS extended transport subsumes all functions and services of basic transport. It would not, therefore, be difficult to extract features of the remaining classes -- 0, 1 and 3 -- from the NBS specification.

SERVICE DESCRIPTION: THE USER'S INTERFACE

The services described below, with the exception of the graceful close, are derived from ISO work [ISO 80a].

Transferring Bulk Data and Small Transactions

All identified applications using transport service require only two distinct data transfer services -- bulk data transfer and 'small' transactions. Transport has a connection-oriented service to transfer, for example, large files, and a connectionless mode to send single-unit transactions. The connection-oriented mode can be loosely equated to a network virtual circuit wherein a logical connection is established and maintained for the duration of the data transfer. Similarly, the connectionless mode can be likened to a datagram network wherein a unit of data is transmitted as a single event without the overhead of establishing, maintaining, and terminating a logical connection.

The connectionless service data unit is expected to be 'small', although there is no length restriction. The connection-oriented t-user may decompose the bulk data into segments called transport-service-data-units (TSDUs). The TSDUs may vary in length and their length is not restricted. Figures 3 and 4 picture the t-user's view of these modes of data sending.

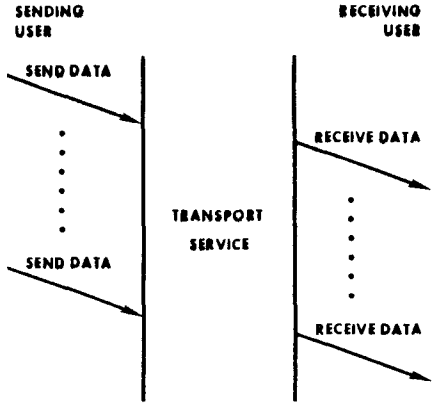

Fig. 3 Bulk data transfer service events

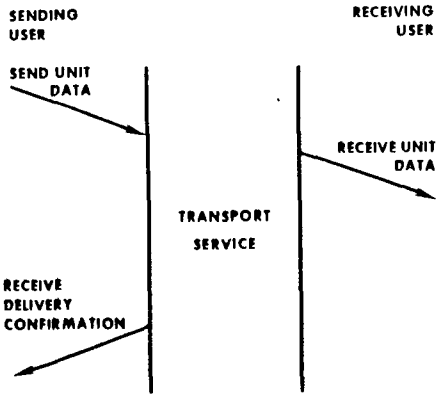

Fig. 4 Transaction service events

Signalling and Interrupting

Expedited data service provides a signalling or interrupt capability, within the connection-oriented mode, such as the ability to send a terminal break character to a remote application. The data is expedited in that it is not subject to the normal flow control mechanism governing data transmission over the connection. The t-user may send up to 16 octects of expedited data. Use of the expedited service is restricted to a single outstanding unit. That is, transport may not send a subsequent expedited data unit over a given connection until the previous one has been received and acknowledged by the receiving transport.

Getting Status Service

The connection-oriented t-user may request and receive status or may receive unsolicited status about the transport connection. The meanings of requests and replies are implementation-dependent and generally local in nature, and thus are left unspecified in the standard.

Getting a Connection

Supplementary to the data transfer service, in the connection-oriented mode, transport must manage connections -- establish, maintain, and terminate them. From the t-user's view, establishment appears through two service primitives; one initiates a connection and the other accepts or rejects the connection attempt. The t-user initiating the connection may send a small amount of data to its correspondent along with the establishment request.

A number of parameters attend the connection request and thus pass between the t-user and transport. Syntax and the exact semantics are not specified in the standard, since these are implementation-dependent. However, the standard specifies the presence of these services. A brief description of these parameters follows.

The t-user must identify itself and its correspondent to transport. The addressing scheme is implementation-dependent, but transport must be able to map a component of the correspondent's address into the network location of the peer transport.

Optionally, the t-user may also specify the relative priority with which data will be transmitted over the connection.

A level of security afforded by the connection and an allowable recipient user group identification may optionally be specified. It is expected that organizations following the standard will specify their own exacting requirements, as an auxiliary component of transport, to assure security level and user group use and conformance.

The class of service, described in the previous section, may also be selected.

In using the connectionless service these same parameters of connection establishment are present, along with the single TSDU. There are minor variations. Three levels of service are provided: basic, resulting in the use of the basic class protocol, and two options of extended class, which differ in their reliability and cost. With both options the t-user of connectionless service inserts a TSDU and receives a reply as to the successfulness of delivery. Using the first option, the peer transports actually exchange two messages to complete the transaction, whereas using the second option, the peer transports exchange three messages to render the service. The latter is more reliable but more costly than the former.

In addition to the connection establishment-associated parameters, transport reports (through parameters) to the connectionless user the result of the transaction, and if it failed, why. (Note that, in general, failure to deliver a TSDU can occur as the result of a network failure, remote host failure, or abortion by either peer transport.)

Sending and Receiving Data

The second phase of connection management, maintaining an established connection, is seen by the t-user only in terms of sending and receiving complete TSDUs. Connections, once established, are full duplex, i.e., two-way simultaneous. Thus, either of the connected t-users may both send and receive TSDUs. Also, they may send and receive asynchronously, without regard to turn or token passing as in half duplex operation.

Releasing a Connection

The third phase of connection management is connection termination. There are two ways to disconnect. The normal release, called 'graceful close', requires the agreement of both t-users. When one t-user has finished sending TSDUs it issues a graceful close request to transport. If the corresponding t-user has more data to send or there is outstanding data that has been sent but not acknowledged, the remaining data will be sent and acknowledged before a close is

returned by the peer transport. The peer transports properly manage this sequencing, including the case where close requests collide, so that no data is lost or unsent. All TSDUs sent in either direction are delivered in sequence to the appropriate t-user before the connection is terminated. The t-users receive confirmation that the connection is terminated.

The other way to terminate a connection is immediate. Either t-user may request that the connection be immediately severed. Any pending or unsent data destined for the t-user requesting the disconnect is not delivered. Data sent, by the t-user requesting disconnect, just prior to the request may or may not reach its correspondent destination. Because of the possible loss of data, the abrupt disconnect termination request is expected to be used only under abnormal conditions, such as a fail-soft condition arrising in one of the hosts, or else where the equivalent of graceful close is accomplished by a higher layer protocol.

Multiplexing

Transport may select multiplexing in both basic and extended classes. Sharing an establtendshed network connection by t-users tends to optimize costs.

Multiplexing of a single t-user connection onto multiple network connections to potentially increase throughput is not done by transport. To potentially increase throughput, the t-user may equivalently construct multiple connections to its correspondent, see Fig. 5.

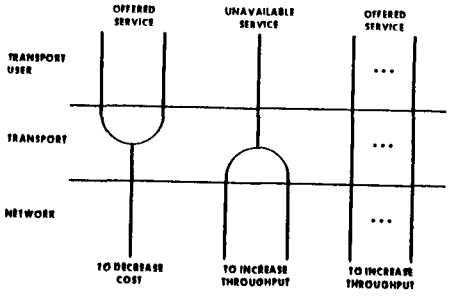

Fig. 5 Distinction between cost-related and throughput related multiplexing

Service Summary

The transport service consists of connection-oriented and connectionless data transfer. Data is delivered reliably and in sequence. Connection status may be obtained upon request. A small amount of urgent data may be sent out-of-band with respect to the normal data flow control. Usually, a connection is closed by mutual transport agreement after all data has been sent and received, but abrupt termination by a transport user is possible when needed.

PROTOCOL FUNCTIONS

The protocol functions described below, with the exception of the graceful close, are derived from the most recent ISO work [ISO 80b].

Connection Identification

During establishment, each peer transport assigns a reference number to its end of a connection. The pair of reference numbers allow the peer transports to uniquely identify the connection for purposes of transmitting data and detecting duplicate connection requests.

Peer Negotiation and Parameter Certification

The unit data size exchanged by peer transports (as opposed to t-users) is determined during initial handshakes as follows. The initiating transport proposes a unit size to be used over the connection. The peer sends back a size. The smaller of the two sizes, if they differ, is used.

Other parameters are checked or negotiated between peers. These include service class, the version number of the protocol, security level, and user group identification.

Connection Establishment and Termination

A two-way message exchange, request and accept, is used by the peer transports in the basic class to establish a connection. Data from the initiating t-user and accompanying the connection request can be delivered by the receiving transport to the t-user's correspondent at the time the t-user is informed of the connection attempt. Because the extended class does not assume inherent network reliability, it uses a three-way message exchange: request, accept, and acknowledgement-of-the-accept. Data

accompanying the connection request can be delivered to its correspondent as in basic class. Figure 6 illustrates connection establishment.

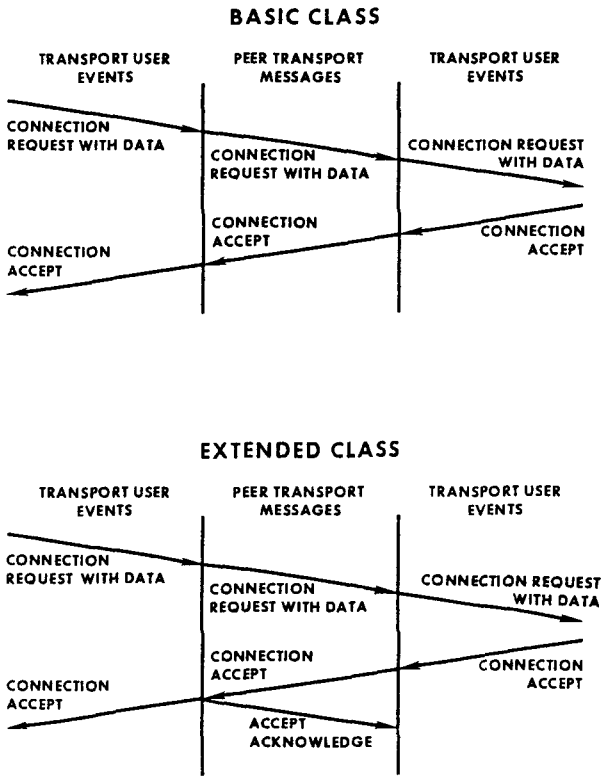

Fig. 6 Connection establishment

The abrupt disconnect service is implemented by peer transports by their exchange of disconnect requests and confirms, whereupon the transport connection is terminated with possible loss of data. The graceful close is implemented by exchanging four messages -- a request and acknowledgement by each transport. The way graceful close avoids loss of data in transit is by placing the graceful close protocol-data-unit (PDU) in the regular data sequence space. (Data PDUs carry sequence numbers so that their order may be preserved.) Thus, if the graceful close arrives before some data unit sent earlier, it will be held and not acknowledged until the data PDUs arrive to fill in the gap in the sequence space. (The sequence space is explained in more detail later.) Unlike the abrupt disconnect, a graceful close will not be returned by the peer transport until all of the user's data has been sent and acknowledged.

Data Transfer

The unit of data transmitted between peer transports is called a transport-protocol-data-unit (TPDU or PDU). The size of the PDU is negotiated by the peer transports during connection establishment. Recall that the transport-service-data-unit (TSDU) passed between the t-user and the transport is variable in length. Generally, a TSDU must be fragmented by the transport into a sequence of PDUs of common size, except the last which is usually shorter. If fragmented, the receiving transport must reassemble the PDUs and deliver a complete TSDU to its user. Each PDU to be transmitted is given a consecutively assigned sequence number serving several purposes. In general, the sequence number controls the flow of data over the connection. In the extended class the sequence number is used to reorder PDUs arriving out of sequence and to detect duplicates.

Expedited data is sent in a special PDU over the existing connection, but it is not subject to the normal flow control. Specifically, this PDU may be sent when normal data may not be sent, and is guaranteed to be accepted by the peer transport. To allow proper buffer management by transport, then, only one such PDU may be outstanding. That is, a t-user may not send a subsequent expedited data unit until the previous one has been acknowledged. The expedited data unit is limited to a maximum of 16 octets of t-user data. Transport engages separate (from the normal data PDUs) retransmission timers to monitor the progress of expedited data.

Transactions may be sent using the connectionless service. Peer transports exchange two messages in basic and extended class to complete the transaction. Optionally, for greater reliability at increased cost and using the extended class, the transaction may involve a three-way message exchange between transport peers as earlier explained under transport services. Since transactions are not restricted in length, the receiving transport may not be able to accept the entire data in one peer message. If not, then unbeknown to the t-user, the peer transports establish the logical equivalent of a connection to transmit and receive the data, then close the pseudo connection.

Flow Control and Receipt Confirmation

The peer transports regulate the flow of data exchange between themselves through the use of a window mechanism. Using the PDU sequence numbers described earlier, each transport informs the other of its current window size (or credit); in effect, the number of PDUs that it is able to accept from its peer. This flow control mechanism has the dual purpose of maintaining data flow while avoiding buffer overflow.

PDUs received are positively acknowledged by sending an acknowledgement PDU to the peer transport. The acknowledgement contains the sequence number of the next expected PDU and thus acknowledges receipt of all PDUs of lower sequence numbers.

The peer transports continually (based on an inactivity timer) exchange acknowledgement PDUs, even when they are not confirming PDUs received. This serves to update the credit and maintain data flow over the connection.

Reliability Assurance

The extended class is aimed at a possibly unreliable network, or at concatenated networks where each network may be fairly reliable but in tandem may not be. Therefore, extended error class controls protect against data loss, replication, out-of-sequence or damaged PDUs, and detect a broken network or remote host.

Lost data is detected by the sending transport when it does not receive acknowledgements. If acknowledgements are not received within an acceptable time interval the PDU is retransmitted. Transport clocks the expected response time by using retransmission timers.

Duplicate PDUs are simply acknowledged and then discarded by the receiving transport.

PDU sequence numbers (part of the peer transport control information) allow transport to find gaps in the succession of incoming PDUs. By checking sequence numbers, the receiving transport can detect and buffer any data arriving ahead of other data which should logically precede it in the TSDU. Transport, then, reorders the PDUs during reassembly of the TSDU before delivery to its t-user.

Data errors are detected by checksumming the entire PDU. Transport uses the 16-bit 1's complement of the 1's complement sum of all double octets in the PDU. Damaged data, detected by the receiving transport, are discarded. Recovery is accomplished by the sending transport in the same manner in which it recovers from lost PDUs.

A remote host that has crashed, a network failure resulting in a broken (but unsignalled) network connection, and deadlock due to peer transport misunderstanding of flow control credit are all resolved by transport through the use of inactivity timers. After a measured time interval transport sends its peer an acknowledgement containing flow control credit information. This resolves some deadlocks. After a sufficient number of retries without response, a hard failure (broken network or host) is assumed and transport closes the connection.

NETWORK INTERFACES

Transport is independent of the characteristics of the underlying networks. The single specification of transport [BURJ 81] provides for operation of either basic or extended class over a variety of packet-switched networks, including commonly existing virtual circuit and datagram networks. For transport to stand alone, the network layer of the architecture is conceived as having two parts, one of which maps from transport to a particular network. The bottom part of the network layer defines a specific communication service to be connected to the standard transport. The top part of the network layer, called the network interface sublayer (NISL) connects transport to a particular communication service. Thus, NISLs differ from one another, are implementation-dependent, are network dependent, and are not part of the transport standard. Burruss, et al. [BURJ 81] specify, as examples, an interface to an X.25 virtual circuit network [CCITT 79] and an interface to a datagram network [IP 80]. Figure 7 illustrates this concept.

The services provided to the standard transport by a NISL are similar to, but simpler than, those provided by transport to the t-user. Namely, a connection request/accept and disconnect make and break network connections. Status service is included. Data send and data receive complete the necessary services.

When interfacing to an X.25 virtual circuit network, the NISL must have packet-level procedures for virtual call service, data transfer, flow control, and restart. When interfacing to a datagram network, the NISL must have procedures for data transfer and for accepting error indications by the network, upon packet loss. (Note that not all lost data will

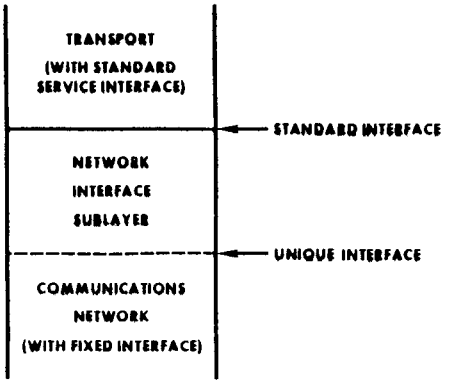

Fig. 7 Coupling transport to a particular communications service.

be detected and reported by the datagram network.)

In summary, the transport interfaces to a unique communication service through a network interface sublayer. This sublayer allows the same transport to be used over different kinds of communication networks. The primitives of this sublayer are few and simple.

SYSTEM INTERFACE

Timer Service

Transport requires but one service of the system, a timer service. (Other needed resources such as memory management, interprocess communication, and task scheduling are implementation-dependent and local in nature, and thus are omitted from the standard specification.) The timer service is obtained through three system primitives. One primitive, initiated by transport, requests that the system start a timer, and provides the time increment and a counter. The counter is untouched by the system; the system merely holds it and returns it upon timer expiration. Counters are incremented and reset by transport and are generally used to tally some maximum permissible number of events, such as the number of retries for retransmission of a data PDU. A second primitive, initiated by the system, indicates to transport that the specified time has elapsed. The timer's identity is returned along with the associated counter. The third primitive, from transport to the system, requests that a timer be cancelled; the parameter of the primitive identifies the timer.

Basic Class Timer

Transport uses timers to maintain the connection and guarantee data integrity and delivery. Since the basic class does not significantly enhance the reliability of the network service, it has no need for a variety of error control timers. Basic class uses only a flow control timer. It is set to indicate a closed window associated with a connection. When the timer expires, if buffer resources permit accepting more PDUs, an acknowledgement is sent to the peer transport issuing a new credit field.

Extended Class Timers

The extended class operates over networks that may fail without signalling transport. It is designed to detect and recover from broken connections, data loss, duplication, and missequence. Several types of timers (each associated with each connection) are required to support this service enhancement.

The window timer triggers the sending of an acknowledgement PDU which synchronizes and updates flow control information.

Inactivity timers allow transport to discover external failure such as a remote system crash or a broken network connection. The timer is reset each time a PDU is received from the peer transport. If it expires, transport closes the connection. The timer's value should be somewhat longer than that of the window timer to ensure that a valid connection is not closed.

A retransmission timer is set for each outstanding PDU. If a timer expires before an acknowledgement arrives, the PDU is retransmitted and the timer's counter is incremented. If the count reaches the allowable maximum value then a give-up timer is set.

Give-up timers measure the maximum time that the transport will wait after overflow of the associated retransmission timer's counter. The time periods of retransmission timers are based on expected response times, whereas the time periods of give-up timers are calculated based upon the maximum packet lifetime within the network and the time required by the peer transport to generate an acknowledgement. When a give-up timer expires transport closes the connection.

Reference timers measure the period of time during which reference numbers are unavailable for reuse in establishing a new connection. When a connection is closed, the reference number must be temporarily suspended from use. If not, the reference number might cause confusion in a new connection with old duplicate PDUs that may still exist in the network.

In summary, transport uses the system clock or other system timing or polling devices to time the inactivity and the interval between successive retries when acknowledgements are not forthcoming from the peer transport. Timer facilities to set, cancel, and awaken transport upon their expiration are provided through system primitives.

CONCLUDING REMARKS

NBS has defined a transport protocol based on the work of ISO and consistent with the needs of the agencies of the U.S. Government. The NBS specification extends the ISO work in that the NBS specification provides a complete design specification that can be referenced in procurements and implemented by system providers. Any organization desiring to procure or build a network must develop such a specification or, alternatively, use the NBS specification. If enough organizations use this specification in their network procurements, then efficient off-the-shelf solutions should become readily available. In addition to this specification, NBS has available an implementation in the C language, and PASCAL program segments which are part of the formal machine specification. In the near future, based on the work within ISO, NBS will develop draft standards for internetwork, session, file transfer, virtual terminal, and remote job entry. This set of standards, if approved, will provide a basic structure to support more advanced distributed systems in the future.

References

[BURJ 81] Burruss, John, Gregory Pearson, Ross Callon, Thomas Blumer and Richard Tenney, "Specification of the Transport Protocol," prepared by Bolt Beranek and Newman, Inc., for the National Bureau of Standards, Technology Building, Room B212, Washington, D.C. 20234, 1981.

[CCITT 79] Revised Recommendation X.25 Preface and Level 3, Study Group VII contribution No. 384, International Telephone and Telegraph Consultative Committee, August 1979.

[ICCC 80] Blanc, Robert P., and John F. Heafner, "The NBS Program in Computer Network Protocol Standards," Proceedings of the Fifth International Conference on Computer Communication, Atlanta, 27-30 October 1980.

[IP 80] Department of Defense Standard Internet Protocol, Defense Advanced Research Projects Agency, January 1980.

[ISO 80] Data Processing - Open Systems Interconnection - Basic Reference Model, Draft Proposal ISO/DP 7498, December 3, 1980.

[ISO 80a] Draft Transport Service Specification, International Organization for Standardization, ISO/TC97/SC16 N563, November 1980.

[ISO 80b] Current Work on Draft Transport Protocol, ISO/TC97/SC16 Ad hoc working group on transport, TB12, November 1980.

[TCP 80] Department of Defense Standard Transmission Control Protocol, Defense Advanced Research Projects Agency, January 1980.