ECMA

EUROPEAN COMPUTER MANUFACTURERS ASSOCIATION

OPEN SYSTEMS INTERCONNECTION

DISTRIBUTED INTERACTIVE PROCESSING ENVIRONMENT (DIPE)

TR/29

September 1985

ECMA

EUROPEAN COMPUTER MANUFACTURERS ASSOCIATION

OPEN SYSTEMS INTERCONNECTION

DISTRIBUTED INTERACTIVE PROCESSING ENVIRONMENT (DIPE)

TR/29

September 1985

TABLE OF CONTENTS

TABLE OF CONTENTS (cont'd)

1. GENERAL

1.1 Introduction

This Technical Report, ECMA TR/29, is one of a series of documents in which ECMA presents the results of its work on distributed applications. The purpose of this Technical Report is to describe and explain the first results of ECMA work on OSI distributed-interactive-processing. A subsequent Technical Report will present ECMA work on Remote Operations (see CCITT Rec. X.410).

ECMA has investigated the characteristics required for distributed-interactive-processing. Some of these characteristics do not yet exist in OSI standardization. The study included currently available specifications from different manufacturers and different standards bodies, which represent the current state of the art. This has led to the recommendation that a special "environment" is required. This provides the structures and services to be used by distributed-interactive-processing applications. It is named the Distributed Interactive Processing Environment (DIPE).

The ISO 7498 layering is used. The focus is the upper layers: Application Layer, Presentation Layer and Session Layer. ECMA believes that this combination of OSI communications and interactive processing technology will provide a good basis for customers' and manufacturers' investment.

1.2 Scope

The subject of this Technical Report, ECMA/TR29, is a Distributed Interactive Processing Environment (DIPE) which supports distributed-transaction-processing. Some possible approaches to providing such an environment have been examined, and proposals are made as follows:

- a statement of the requirements for DIPE (see 2),
- a technical overview of DIPE (see 3),
- a DIPE service description (see 4),
- a list of items for further study (see 5),
- recommendations for future standardization work (see 6).

The DIPE field of applications includes, but is not limited to, OSI distributed-transaction-processing for: office information systems, remote database-access, specific value-added-network (VAN) services, and OSI management. DIPE may also be suitable for use in many non-interactive areas of OSI.

1.3 References

ISO 7498	: Data Communications, Open Systems Interconnection, Basic Reference Model

ISO 8326	: Information Processing Systems, Open Systems Interconnection, Basic Connection-Oriented Session Service Definition
ISO 8327	: Information Processing Systems, Open Systems Interconnection, Basic Connection-oriented Session Protocol Specification
ISO 8509	: Information Processing Systems, Open Systems Interconnection, Service Conventions
ISO 8649	: Common Application Service Elements
CCITT Rec. X.410	: Message Handling Systems, Remote Operations and Reliable Transfer Service
ECMA-	: Distributed Applications for Message Interchange (MIDA), Message Transfer Service Access *(under development)*
ECMA-	: OSI Directory, Concepts and Directory Access Service *(under development)*
IBM/1	: Systems Network Architecture, Transaction Programmer's Reference Manual for LU Type 6.2
Xerox/1	: Courier, The Remote Procedure Call Protocol. Xerox System Integration Standard XIS-038112, Xerox Corporation, Stamford, Connecticut, USA.

1.4 Definitions

For the purpose of this Technical Report the following definitions apply.

1.4.1 General Terminology

The following terms are used with the meanings defined in ISO 7498:

- open-systems-interconnection
- service, layer service
- service-user, service-provider
- service-data-unit
- protocol, layer protocol
- connection-oriented data transfer
- connectionless data transfer.

The ISO 8509 terminology is used in the service description (see 4.1.2).

1.4.2 Specific Terminology

1.4.2.1 processing: the execution of computer programs.

1.4.2.2 distributed: dispersed and separated with respect to various criteria (e.g. in separate computers).

1.4.2.3 distributed-processing: co-operative execution of computer programs which are dispersed into separate computers.

1.4.2.4 <u>interactive</u>: depending on the transfer of information between two or more participants.

1.4.2.5 <u>interactive-processing</u>: processing which depends on the transfer of information between two or more participants during execution.

1.4.2.6 <u>distributed-interactive-processing</u>: processing with the combined characteristics of distributed-processing and interactive-processing.

1.4.2.7 <u>processing-environment</u>: that which provides the resources used to execute computer programs, and manages the resources and the execution.

1.4.2.8 <u>distributed-interactive-processing-environment</u>: a processing-environment which supports distributed-interactive-processing.

1.4.2.9 <u>OSI-distributed-interactive-processing-environment</u>: a distributed-interactive-processing-environment which conforms with relevant open-systems-interconnection standards.

1.4.2.10 <u>DIPE-service</u>: the set of open systems interconnection services which support OSI-distributed-interactive-processing.

1.4.2.11 <u>transaction</u>: a defined and logically complete unit of work which is interactive.

1.4.2.12 <u>transaction-processing</u>: processing of transactions.

1.4.2.13 <u>distributed-transaction-processing</u>: transaction execution dispersed into logically separate computers.

1.4.2.14 <u>DIPE-program</u>: any program which executes in the distributed-interactive-processing-environment.

1.4.2.15 <u>DIPE-processing-tree</u>: an abstraction which defines distributed-interactive-processing in terms of interconnected DIPE-program instances.

1.4.2.16 <u>DIPE-dialogue</u>: the communication which occurs between DIPE-program instances via the DIPE-service.

1.4.2.17 <u>DIPE-node</u>: the provider of the distributed-interactive-processing-environment to logically co-located DIPE-programs.

1.4.2.18 <u>DIPE-network-arc</u>: an abstraction which defines pairs of DIPE-nodes which support distributed-interactive-processing between DIPE-programs.

1.4.2.19 <u>logical-network</u>: any network which is defined without reference to its physical realization.

1.4.2.20 <u>DIPE-logical-network</u>: a logical network which consists of DIPE-nodes and DIPE-network-arcs.

1.4.2.21 DIPE-program-name: the identifier used to reference a DIPE program.

1.4.2.22 DIPE-node-name: the identifier used to reference a DIPE-node.

1.4.3 Acronyms

OSI	open system interconnection
DIP	distributed interactive processing
DIPE	distributed interactive processing environment
CASE	common application service elements defined in ISO 8649
VAN	value-added network
SDU	service data unit
QOS	quality of service
ROS	Remote Operations Service, defined in CCITT X.410

2. DIPE REQUIREMENTS STATEMENT

2.1 Rationale

The field of application of this Technical Report is primarily high-performance distributed transaction processing. Some consequences are:

- Functionality. The OSI standards needed for this functionality extend beyond data communications and into some aspects of distributed-processing structure, distributed-applications design and possibly program interfaces.

- Qualities. Transaction processing makes a very special demand on performance efficiency, reliability, resilience, recovery, security and privacy characteristics.

It should be no surprise that the early OSI protocols emerging from the standardization process need extensions for this particular field.

The proposed Distributed Interactive Processing Environment (DIPE) standardization is designed for OSI success in this field.

2.2 Policy Requirements

The policies, which define the way in which the DIPE standardization process is required to operate are as follows:

P1 OSI Acceptability. DIPE standardization shall achieve effective use of, and evolutionary progression of, existing OSI work. This may include engineering exercises for selection of options in the underlying layers, to best meet DIPE needs.

P2 User Oriented. DIPE standardization shall satisfy user needs, and shall be understandable from user viewpoints which may be somewhere outside the OSI Reference Model.

DIPE structure should be designed by starting from the corresponding top-down viewpoint; specifically, that of the designers and administrators of distributed applications, not that of the communications engineer.

P3 <u>Application needs</u>. DIPE services shall initially be specified to support transaction processing between OSI applications. The DIPE design should be capable of future extension to meet other distributed interactive processing needs.

P4 <u>Constraints and generality</u>. To satisfy the demanding quality attribute requirements below, DIPE standards shall, where necessary, constrain the allowed behaviour of the distributed applications which use them. A complementary aim is to support the widest possible range of uses of DIPE standards, restricted only by these necessary constraints.

P5 <u>ECMA practical experience</u>. There is no single user as a source of requirements for the full range of DIPE services. Therefore, the detailed requirements should be strongly based on the practical implementation experience of ECMA Member Companies, and on existing proven designs available to them.

P6 <u>Structure</u>. DIPE standards should provide structured solutions to the needs of DIPE users and implementors. These should include universal basic services, with option sets of extended services.

P7 <u>Variety reduction</u>. Variety reduction is a major aim. DIPE standards are required to avoid proliferation of options and alternatives in ways which would harm interoperability.

2.3 <u>Function Requirements</u>

The functional characteristics required from standards in the DIPE field of application are as follows.

F1 <u>Processing Environment</u>. Define a Distributed Interactive Processing Environment to be provided by OSI standards, and used by OSI distributed applications.

F2 <u>OSI Services</u>. Define the abstract OSI service which models the external protocol interactions between application-entities which are located in separate OSI end-systems.

F3 <u>OSI Protocols</u>. Define the OSI protocols to provide the OSI services required in F2.

F4 <u>Interface Definition</u>. Define an interface for interactions between the components of distributed applications. The characteristics are to be such that:

(a) The interface is easy for application designers and implementers to use correctly and efficiently.

(b) The interface can be realized consistently and with assured inter-operability in different programming languages and different program execution environments.

(c) The interface preserves the invariant properties necessary for portability of applications software design between different realizations of it.

F5 Management. Define the concepts and technical structure for the management and administration of DIPE. Define the related usage of OSI management protocols.

NOTE

Requirement F4 raises important standards policy issues. These are reviewed in section 3.7 and are included in section 5 for further study.

2.4 Quality Attribute Requirements

The qualitative characteristics required from standards in the DIPE field of application all have a common theme: excellence.

Achievement of these high quality attribute levels is very demanding. They will necessarily drive the design trade-offs in DIPE standards.

This is an "intercept" requirement. Over time, the target levels for excellence will move. The cycle time is several years for standards development and consequent product development. Therefore, to satisfy these requirements, DIPE standardizers need to predict likely future competitive quality attribute levels, agree them, and design the standards accordingly.

The quality attributes required in DIPE standards are as follows:

Q1 Efficiency. The standards shall be designed to enable their implementations to achieve excellent performance efficiency. Appendix A defines specific measurement methods and targets.

Q2 Reliability. The standards shall be designed to enable their implementations to achieve excellent reliability characteristics.

Q3 Security. The standards shall be designed to enable their implementations to achieve excellent security and privacy characteristics.

The DIPE standards should allow achievement of these specific characteristics (Q1, Q2, Q3), and also have the flexibility for implementations to make their own trade-offs choices between them.

3. TECHNICAL OVERVIEW

3.1 General

This is a technical overview of a proposed OSI distributed interactive processing environment, which will be referred to as DIPE. The description is intended to be understandable to a wide audience of people with general technical knowledge of networked Information Systems. It includes an informal description of the structure and services which are specified in section 4.

To assist the reader, the concepts and terminology used are explained informally in the text. For exact definitions, the reader should refer to section 1.4.

The DIPE protocol and service are within the scope of Open Systems Interconnection standardization (OSI). The ISO 7498 layering is used. The focus is the upper layers: Application layer, Presentation layer, Session layer and the use of Transport services. The proposed protocol and service select from and progress existing ISO and CCITT standards and drafts on Upper Layer Architecture, Common Applications Service Elements (CASE), Remote Operations Service (ROS), OSI Session and Presentation layers, and OSI Management. Section 3.6.1 explains the positioning of DIPE in the OSI layers.

Work complementary to this TR will be included in an ECMA Standard where Remote Operation services will be defined on a basis of CCITT Rec. X.410.

3.2 The Fundamentals of DIPE

The basic function of DIPE is to provide services to support distributed interactive processing.

DIPE provides a combination of selected OSI upper layer services and selected services to support distributed processing. DIPE provides exactly and only the services required for OSI distributed interactive processing standardization. This exclusive focus is an opportunity for unified and coherent structure.

Distributed processing is the necessary starting point for understanding the fundamentals of DIPE.

- Distributed processing is processing in which programs are executed co-operatively in separate computers.

The essential point is that the prime concern of distributed processing is processing, not communications.

Communications protocols are included only because the processing is distributed; they are not fundamental to processing itself. For our OSI standardization purposes, it is vitally important that the communication is actually by OSI protocols between OSI end-systems, and that everything is then seen in the context of the ISO 7498 Reference Model.

But this OSI orientation is not inherent in distributed-processing.

The next step in this orientation is to clarify what are the fundamentals of processing, whether distributed or not.

- Processing is fundamentally concerned with the execution of programs and with managing the provision and consumption of execution resources.
- Execution is the systematic consumption of execution resources to animate the programmed logic of computer programs.
- Execution resources are processor cycles, memory and the information channels via which the processing communicates. They are all serially re-usable.

Execution is concerned with the here-and-now; and not previous processing, and not future processing, neither is it concerned with the structure, purpose or meaning of the programs executed.

This analysis gives a clear separation of concerns.

- DIPE is concerned with co-ordinated immediate execution of whatever distributed transactions are activated at its service boundary by the actions of distributed applications. DIPE is also concerned with managing the orderly provision and consumption of DIPE execution resources.
- DIPE is not concerned with the what, why and how of distributed transactions. They are presented to it as stereotyped interactions at its service boundary, and their logical content is not visible to DIPE. DIPE has no concept of these interactions as logical bindings; or as jobs to be remembered, managed and scheduled; or as activity to be resumed, recovered and re-run after failures.

  These are all concerns of the users of DIPE.

The DIPE services which DIPE is consequently required to provide are simple, but very powerful. See section 3.6.

3.3 DIPE Processing Trees

Programs which achieve distributed processing via use of DIPE services are referred to as DIPE programs. The communication between two DIPE programs via the DIPE service is referred to as a DIPE dialogue. The complete group of interconnected DIPE programs and their DIPE dialogues is collectively referred to as a DIPE processing tree.

Fig. 1 illustrates a simple DIPE processing tree. The arrowhead shows the direction of the establishment of the DIPE dialogue between the programs. A request from A has caused the DIPE service to establish a DIPE dialogue to a fresh instance of B, and to begin execution of B. This direction is not to be confused with that of data transfer, which may be in both directions.

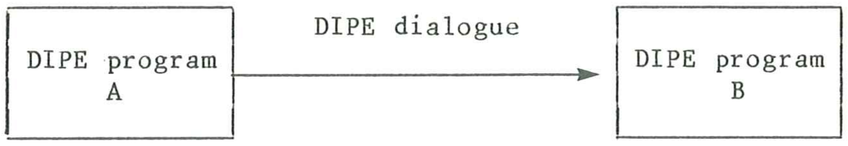


Fig. 1 - Simple DIPE Processing Tree

An essential concept is that the establishment of the DIPE dialogue invokes a "fresh instance" of the destination program, B. The requirements are that the processing of this instance of B is dedicated to the needs of A; that it cannot be accessed by other DIPE dialogue establishment requests; that it is not contaminated by debris from any previous use; and that it has data integrity protection from interference effects of other concurrent processing. This concept also ensures that all DIPE processing trees are separate. If two trees apparently include the same DIPE program identified as x, the processing is by separate instances of x.

The way of providing this "fresh instance" is implementation-dependent (e.g. separate program image, re-entrant code, new process, allocation of an existing process, etc.).

This "fresh instance" concept requires all DIPE programs to have a structure which depends upon their processing being "instantiated" via a DIPE dialogue established from an external source. This applies also to the root programs of a DIPE execution tree (e.g. A in Fig. 1). The way in which the DIPE dialogue into this first program is established is outside the scope of this Technical Report. Typically it would result from an operator load request, workstation start-of-day actions, job scheduling actions, etc.

A more complex example is illustrated in Fig. 2. A DIPE program instance can only have one DIPE dialogue established into it (the one which "instantiates" it); but it may establish as many DIPE dialogues outwards from itself as implementation limits allow. The "fresh instance" concept therefore ensures that a DIPE processing tree always has a tree structure, never a mesh.

The tree structure of the inter-program communication and the mutual exclusion of processing are fundamentally necessary for orderly, reliable, high-integrity distributed processing. These are pre-requisites for use of OSI concurrency/commitment/recovery (GCR) techniques.

At run-time, the configuration of the tree varies over time as the DIPE dialogues are established and terminated. Some trees have a pre-defined and fixed structure; the content and structure of others may vary dynamically according to circumstances.

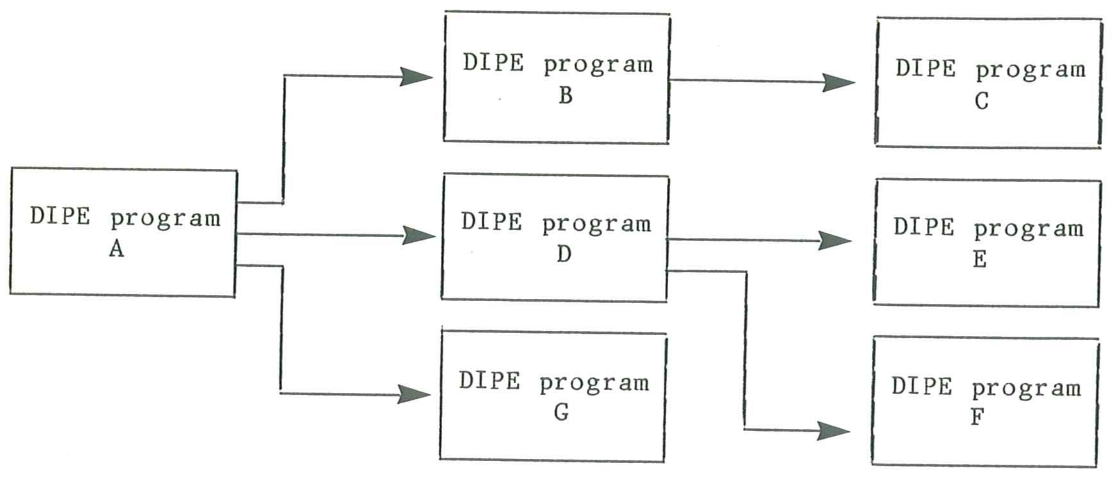


Fig. 2 - DIPE Processing Tree with Multiple DIPE Programs and DIPE Dialogues

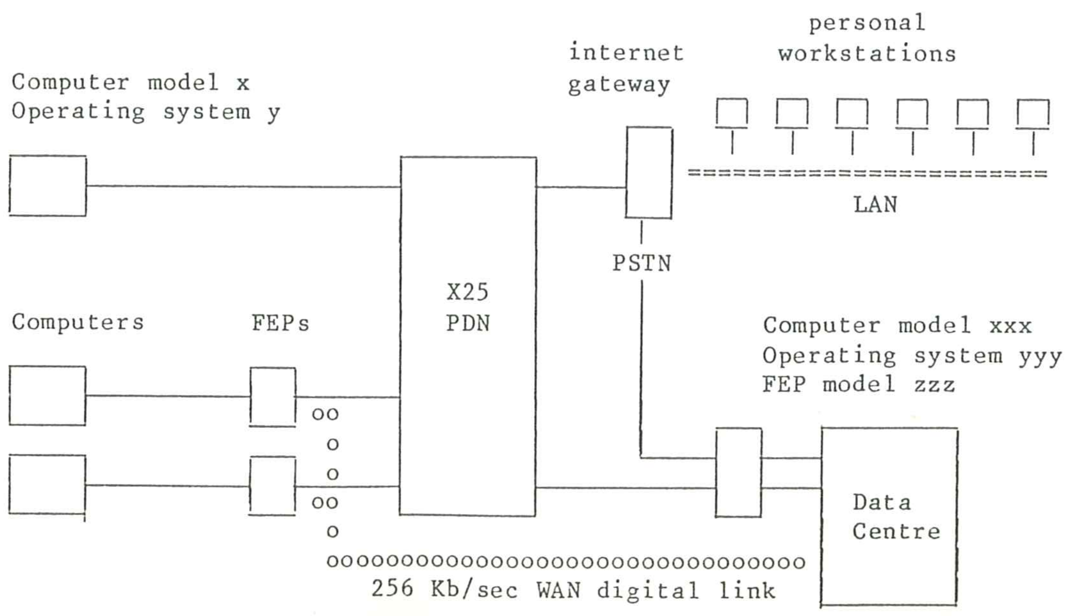
These DIPE processing trees provide a model of DIPE distributed processing structure, which also models the distributed-applications structure using DIPE.

3.4 DIPE Logical Networks

The DIPE service-providers are modelled as DIPE nodes. In the universe of DIPE, real physical networks (consisting of computers and the data communications between them) are abstracted into logical networks called DIPE logical networks. These consist of DIPE nodes and DIPE network arcs between them. The DIPE network arcs are an abstraction for network management purposes; they define which DIPE nodes can support execution of distributed transactions between them. This is potential connectivity, not an actual communications connection.

Fig. 3 illustrates the physical structure of a computer network example. Fig. 4 illustrates a DIPE logical network which might be constructed by the administrators of the computer network.

Fig. 3 - An Example of a Computer Network



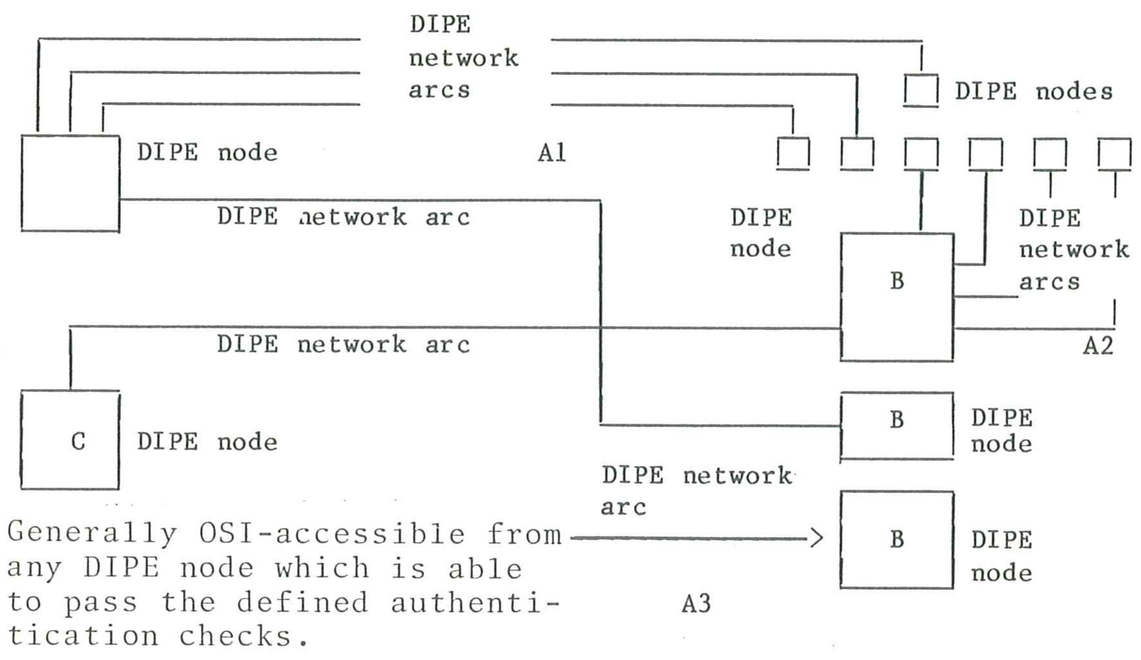


Fig. 4 - A DIPE Logical Network, Modelling the Computer Network Illustrated in Fig. 3

The principle kinds of differences illustrated are:

- Abstraction. There is no visibility of the physical details of the computers, telecommunications media and network topology.
- Controlled connectivity. The allowed DIPE network arcs may be fewer than physically allowed by the actual network topology and internet gateways and routes.
- Network partitioning. The same real network may be partitioned into regions which are separate DIPE logical networks with no DIPE arcs between them (see A1, A2, A3 in Fig. 4).
- Many-to-one. There may be multiple separate DIPE nodes located on the same computer, perhaps providing different kinds of processing services to different kinds of users (see B in Fig. 4).
- One-to-many. There may be one DIPE node spanning multiple computers. For example, a complete distributed processing system or a private network could be defined as a single OSI-visible DIPE node if the administrators wish it to be seen as such (see C in Fig. 4).
- Relocateability. The DIPE nodes are defined and named independently of their physical location. It is logically possible for administrative action to relocate a DIPE node in response to changing circumstances. The processing services which are provided by the DIPE program at that DIPE node move with it.

Most of DIPE management is built around the DIPE logical network structure.

- The DIPE logical network defines the operational scope of all possible DIPE processing trees. Therefore it has a natural structural role in security and access-control.
- The nodes of the DIPE logical network (the DIPE nodes) are the authorities for all execution of DIPE programs. Therefore it has a natural structural role in authentication and accounting.
- The nodes of the DIPE logical network (the DIPE nodes) are the providers of all network-visible serially re-usable execution resources. Therefore it has a natural structural role in resource management, and the avoiding and handling of resource usage conflicts, and the associated interference effects and deadlock/livelock cycles.
- The nodes and arcs of the DIPE logical network (DIPE nodes and DIPE network arcs) are the incremental units for the installation, growth and change of distributed interactive processing capability. Therefore it has a natural structural role in the administration of networked systems.
- Likewise, the DIPE logical network has a natural structural role in the day-to-day operation and management of networked systems.

See also sections 3.6.5 and 4.5.

3.5 DIPE Naming and Addressing Requirements

3.5.1 DIPE naming requirements

The requirements for naming DIPE nodes are that their names should reflect the administrative "ownership" of the nodes, and identify each uniquely within the DIPE logical network in which they operate. The multi-level naming structure supported by the OSI Directory in ECMA-.. *(under development)* is ideal for this purpose. See also section 3.6.1.

The requirements for naming DIPE programs are also relatively simple. Some part of whatever is the corresponding OSI name-space should be reserved for globally unique naming of special DIPE programs (e.g. those with DIPE management roles). The rest of the available name-space should be allocated to meet the needs of DIPE users. See also section 3.6.1.

The destination of all distributed transactions via DIPE can be identified as:

- a DIPE program (where this is generic and unique in the DIPE network scope concerned), or
- a DIPE program at a DIPE node (the DIPE program-name need not be more generally unique).

This allows various levels of DIPE program identification, from the generic to the specific.

There may also be other naming conventions (e.g. local aliases). These would be mapped onto the DIPE names at the DIPE service boundary. This separation allows the naming considerations in the two worlds of DIPE service-users (i.e. distributed applications) and the DIPE service-providers (i.e. distributed processing support) to be structurally independent.

3.5.2 DIPE addressing requirements

The OSI addressing which underlies the DIPE services relates to the world of networked communications. It is not architecturally visible in the DIPE service. The mapping onto the appropriate OSI service-access-points (SAPs) is hidden inside the DIPE service boundary. See also section 3.6.1.

Wherever practicable, the strings of symbols which are DIPE names, OSI SAP identifiers, etc. should have the same (or at least well related) logical structure and syntax, and perhaps related values.

3.6 DIPE Services

3.6.1 Overview and relationship to OSI

The DIPE services are OSI services defined to support the DIPE dialogues and DIPE processing tree structure of distributed interactive processing applications, see 3.3. The services and the applications operate within the distributed interactive processing management structure modelled as DIPE logical networks, see 3.4. Their naming and addressing requirements are as described in 3.5. DIPE programs are the users of the DIPE services. DIPE nodes are the providers of the DIPE services. A DIPE node can be seen as an OSI end-system.

These services should provide only the minimum necessary set of functions, on top of which other richer sets of functions may be built.

The essential run-time functions may be summarized as follows:

- Creation, execution and termination of DIPE program instances.
- Establishment and termination of DIPE dialogues.
- Data transfer via DIPE dialogues, uninterpreted.
- Synchronization of DIPE dialogues and of DIPE program execution across a DIPE processing tree.
- Exception handling across DIPE dialogues.

- Security via universal enforcement of basic access-control and authentication disciplines.
- Accounting via exchange of basic accounting information within the DIPE service.

These need to be supported by resource and configuration management functions and error-reporting and diagnostic functions.

In the ISO 7498 layering, the ultimate users of the DIPE services would be application entities in the Application layer. The DIPE services would be candidates for classification as common application service elements (CASE), see ISO 8649.

The technical characteristics of the DIPE services could be provided by session layer protocol; uninterpreted data exchange via interactions with defined dialogue structure. Presentation layer protocol would format this data and Application layer protocol would add any necessary application control functions (e.g. manipulation of CASE parameters, management of CASE "associations" etc.).

However, ISO 8326 already defines a set of connection-oriented services to be provided by the Session layer, and ISO 8327 defines the corresponding session protocol. The DIPE Connection-oriented Service would be defined as extended Session layer services, supported by some variant of Session layer protocol. This DIPE service functionality could also be constructed from protocol in the Application layer.

The ISO work on connectionless Session layer is at an earlier stage of development. The DIPE Unit-operation Service is a candidate for inclusion in this.

The positioning of DIPE services and DIPE protocol in the OSI layering needs further study.

3.6.2 Connection-oriented Service

Most distributed processing and most OSI standardization is currently based on connection-oriented data transfer. Therefore, DIPE should provide an appropriate connection-oriented service with the required quality attributes.

Existing OSI services and protocols have emphasised open interconnection, but have not included the special characteristics required for distributed transaction processing (see 2.1).

The chosen connection-oriented service is defined in section 4.3 and the related multi-synchronization service is defined in section 4.4.

These services are selected on the basis that they are consistent with distributed transaction processing state-of-the-art, and have the required quality attributes of efficiency, reliability and security. There is correspondence to the IBM SNA LU 6.2 protocol boundary, defined in reference IBM/1 which is publicly available.

The choice of these services satisfies most of the DIPE requirements which are defined in section 2 above.

At this early stage of DIPE work, this choice of connection-oriented services is tentative. Early publication of this Technical Report allows time to consider the issues before proceeding to standardization. They are included in the items for further study.

3.6.3 DIPE Unit-operation Service

The DIPE Unit-operation Service is a specialized lightweight service. It is optimized for efficient support of request/reply transactions such as those in the remote operations service (ROS) defined in CCITT Rec. X.410, Chapter 2, and the Xerox Courier service defined in Xerox/1.

The DIPE Unit-operation Service accommodates remote-procedure-call and message-passing processing structures. It is designed to have very low overheads of communications, processing and memory occupancy; but without sacrificing overall reliability and security attributes. It is defined in section 4.2.

The current ISO work on connectionless session protocol and services does not (yet) include all of the above specialized characteristics needed for DIPE.

A typical use for the DIPE Unit-operation Service is with microprocessor workstations where communication is infrequent and generally involves a relatively small amount of data. However, fast response is essential for transaction processing. The infrequency of communication means that the overheads for maintaining a connection may be unnecessarily large. Also, the resource limitations of microprocessor workstations further reduce the amount of data which can be held and processed for a connection.

The low overheads possible with the Unit-operations Service may also be necessary for multi-user computers. While such computers have more memory and processing resources, the overheads for maintaining connections can still be large. The problem becomes more acute when the multi-user computer is being accessed by users from a number of workstations or other multi-user computers. Maintaining connections to all the computers may cause a high ratio of overheads to productive work.

There are three overheads to maintain a connection:

- frequent handshakes reducing the communications bandwidth;
- processing the handshakes;
- holding connection state information in memory during the periods of inactivity.

The DIPE Unit-operation Service is a connectionless data transfer service to minimize these overheads.

3.6.4 DIPE multi-endpoint services

A multi-endpoint service is one in which a request event at one point should result in indication events at one or more points. The underlying protocol may use broadcast/ multicast communications.

It is expected that multi-endpoint services will be important for implementing OSI distributed systems. The DIPE multi-synchronization service is, in effect, a multi-endpoint service in the DIPE connection-oriented service. It is expected that DIPE Unit-operation Service can be extended to provide multi-endpoint services. This is a topic for further study.

3.6.5 DIPE Management Service

The management characteristics of DIPE logical networks have been described in section 3.4.

It is intended that the DIPE environment be managed in accordance with the management structure defined in ECMA TR/.. *(under development)*. Specific parameter types to be included for DIPE in future management protocols will be studied.

3.7 DIPE User Interface

3.7.1 Introduction

The DIPE Requirements Statement includes a requirement to define and standardize a user interface to the DIPE service (see 2.3, requirement F4).

This is something new in OSI standardization, and raises important policy issues. There are also technical issues, but they are a secondary concern.

Section 3.7.2 explains the current OSI position against interface standardization. Section 3.7.3 explains the rationale for proposing interface standardization. Section 3.7.4 is a preliminary review of technical matters.

A major item for future work for ECMA/TC32/TG2 is to recommend what position ECMA should take on these DIPE policy issues which are here left open. However, the first priority is certainly peer-protocol standardization.

3.7.2 Current position on OSI interface standardization

The current practice in ECMA, ISO and the CCITT is to standardize external protocols (with abstract specifications of the services which they provide and use), but not to standardize the internal interfaces between the software modules which implement these protocols.

The rationale is as follows:

(a) The aim of OSI standardization is homogeneous interconnection between heterogeneous OSI end-systems. This is the essence of "open interconnection".

(b) Therefore, standardization of the external protocol interactions is necessary and sufficient to achieve this aim.

(c) Therefore, standardization of internal interfaces is not necessary for this purpose, and unnecessarily constrains implementation of the protocols.

(d) Moreover, agreeing and achieving homogeneous interconnection is very difficult, and the extra work and controversy and technical difficulty of interface standardization would delay and damage the technical work.

ECMA has always supported this position very strongly. Two further considerations are of particular importance to ECMA Member Companies.

(e) With the standardization of protocols, internal interface design is then, more than ever before, a very important design freedom. Standard internal interfaces would restrict this freedom.

(f) Most networking currently uses various proprietary protocols and interfaces (often established before OSI standardization). Some of these will continue to be used. The OSI standards must co-exist with them.

This requires that existing software structure should be preserved when the use of OSI protocols is added. The likely implementation techniques are to "hide" new protocols inside existing module interfaces, or to use protocol converters or gateways around the outside of existing implementations. A requirement for standard internal interfaces could conflict with this essential freedom.

Any proposal to standardize the internal interfaces for OSI services and protocols must include satisfactory answers to these concerns.

3.7.3 Rationale for DIPE interface standardization

The proposals made here are specific to DIPE. Similar considerations may also apply to other aspects of OSI standardization; but that is outside the scope of this Technical Report.

There are five main reasons for proposing standardization of DIPE interfaces.

1) Needs for improved compatibility and quality attributes via more comprehensive specification of DIPE.

2) Needs for an end-user application-oriented programming interface above DIPE.

3) Needs for a generic description of DIPE which could be used by programming language standardization bodies.

4) Needs for variety reduction in networking software.

5) Needs for applications software portability.

The first four might be satisfied in other ways; but some degree of interface standardization would be helpful (if other considerations do not preclude it).

The fifth need, if accepted, is the decisive one for interface standardization. It is an additional aim to the basic OSI aim in 3.7.2 (a) above. It should be possible for user application designs to be system-independent. The requirement in 2.3 F4 (c) is to provide design portability for applications software. This level of portability is necessary and sufficient for the DIPE interface standardization. It is less difficult technically than code portability, which would be a different requirement.

The concerns expressed in section 3.7.2 might be covered as follows.

- The validity of the current emphasis on protocol standardization (a,b,c,d) is recognized. Any interface standardization is an extra and logically separable item. OSI standardization is also now at a more mature stage at which there is less concern that these additional considerations would damage the OSI work (d).
- The basic freedoms (e and f) could be preserved by separating the conformance requirements for protocols and interfaces. Interface should be defined in separate standards. Their use should not be required for conformance to protocol standards.

3.7.4 Technical proposals

The interfaces being considered for standardization are those between the users of DIPE services and the providers of DIPE services. They would probably have at least the following subdivisions:

- distinctions between the DIPE connection-oriented service and the DIPE Unit-operation service (including the possibility of building the latter on top of the former);
- differentiation between different option sets (see section 4.6).

To avoid unnecessary re-invention and mistakes, the interfaces should be based on existing proven designs.

For the DIPE Connection-oriented service (including the DIPE Multi-synchronization service) the only interface design considered so far is the IBM SNA LU6.2 protocol boundary interface. This is consistent with the proposed DIPE services, and with the considerations above and in section 3.7.3. Other proposals are invited.

For the DIPE Unit-operation service no specific interface proposal has yet been made. One possible approach is to define a standard interface for remote procedure calls. Another possibility is a standard interface for message passing. Probably both styles of use would be built on a common interface. The design principles for these simple and basic mechanisms are well researched, and there is a variety of proven designs to select from. Specific proposals are invited.

4. DIPE SERVICE DESCRIPTION

4.1 General

4.1.1 Introduction

This section gives a detailed description of the proposed DIPE services. Section 3 above provides an overview. These proposals are made here as the basis for further study, which may result in an ECMA protocol standard.

The services are described in four groups:

- DIPE Unit-operation service (see 4.2),
- DIPE Connection-oriented service (see 4.3),
- DIPE Multi-synchronization service (see 4.4),
- DIPE Management service (see 4.5).

Option sets are described in section 4.6. Some characteristics common to the services are described in sections 4.1.2 and 4.1.3 below.

At this stage of the work, it is not confirmed that all these services are needed, nor that they are complete.

4.1.2 Notation

The ISO 8509 notation is used to define the DIPE services.

The structure and parameters of each service are defined in tables with entries showing the relationships between service events and parameters. The table entries are:

X if the parameter is always present in an event (and may be different from the value in any preceding event). The parameter value may be a null, or a default value may be supplied by the service.

(X) if the parameter is sometimes present in an event (and may be different from the value in any preceding event). The parameter value may be a null or default value.

= if the parameter in an event has the same value as in the preceding event in the table (but not necessarily the same representation).

- if the parameter is not present in an event.

The headings "R", "I", "R", "C" are used in these tables for "request", "indication", "response", "confirmation".

4.1.3 Common parameters

This section defines parameters which are common to the Unit-operation and Connection-oriented services.

4.1.3.1 Destination-reference parameter

The DIPE destination-reference consists of the DIPE node name and DIPE program name which are the intended destination of the DIPE dialogue.

NOTE

The DIPE service providers will map this reference into the appropriate OSI layer service access points. Details are for further study. Implementation interfaces may also use local references as substitutes for this reference.

The DIPE node name has the multi-level name structure supported by the OSI Directory Protocol defined in ECMA-.. *(under development)*.

The DIPE program name is taken from a large namespace which can serve several different purposes as follows. (These distinctions are not significant to the DIPE service providers).

a) Normal static DIPE programs. These have roles defined within some non-global scope (e.g. within the networks of a particular business enterprise). Their identifiers need only be unique within the context of the DIPE node name(s) concerned. These DIPE programs typically have a long lifetime.

b) Dynamic DIPE programs. These have any required role as in a) above, but have a restricted (and possibly short) lifetime. Their identifiers are therefore current for that defined lifetime, and invalid otherwise. They need only be unique within the context of the DIPE node name(s) concerned. This kind of identifier allows powerful and flexible use of concepts of mutual exclusion of access, context preservation and binding currency, in ways de-coupled from the DIPE service. This is the key to many important systems characteristics.

c) Special DIPE programs. These have globally defined roles. They might include DIPE management roles, providers of higher level DIPE services, generic OSI service roles and proprietary system roles, etc. Their identity would be "well known" and globally unique within the DIPE logical network scope.

The detailed structure and allocation of this namespace is outside the scope of this TR. A suggested minimum identifier size is 8 octets (64 bits) with variable length up to 64 octets.

There are many technical issues in this area which need further study.

4.1.3.2 Source-reference parameter

This is an identifier provided in the indication event which begins a DIPE dialogue. It identifies the DIPE node at which the request was made. It does not identify the requesting DIPE program. This is a basic security feature; see also 4.1.3.3.

4.1.3.3 Authentication-reference parameter

The authentication-reference parameter defines the security information to be used by the destination DIPE node to authenticate access to the destination DIPE program. It may include a registered user identifier, which the destination DIPE node may use for accounting and audit purposes.

There are two cases:

- One case is that the DIPE service provides a default value by using the security information which it already knows from the request which originally initiated execution of the DIPE program which is now the requestor.

- The other case is where other security information is to be used, and is explicitly provided by the DIPE service user via this parameter.

The nature of this security information is for future study.

This structure includes a restriction which is of fundamental importance for security:

- This security information is used inside the DIPE service, and is not passed to the destination DIPE program (i.e. service user). The need to trust the DIPE service providers not to misuse these passwords etc. is unavoidable; but there is no need to extend this trust to the destination DIPE program.

However, the DIPE service users may exchange the same or additional security control information via their data transfers, which are not interpreted by the DIPE service.

NOTE

Further study is needed to ensure that DIPE has the required security attributes. Contributions are invited.

4.1.3.4 Error types for termination of a DIPE dialogue by the service providers

It is proposed that the error types parameter identifies one of the following reasons for the service provider terminating the dialogue. Further details may be recorded in error logs, see 4.1.3.7.

DIPE dialogue rejected:

because of a permanent problem in the underlying service.

DIPE dialogue rejected:

because of a temporary lack of resources.

DIPE dialogue rejected:

by the remote service provider because he does not support this dialogue profile.

DIPE dialogue rejected:

by the remote service provider because the remote service user does not support the use of initialization parameters (see 4.3.3.3).

DIPE dialogue rejected:

by the remote service provider because the quantity of initialization parameters (see 4.3.3.3) does not match that of the remote service user.

DIPE dialogue rejected:

by the remote service provider because he considers the authentication information (see 4.1.3.3) to be invalid.

DIPE dialogue rejected:

by the remote service provider because the dialogue profile (see 4.3.3.1) in the request included the DIPE-MULTISYNCHRONIZE service which the remote service provider does not support.

<u>DIPE dialogue rejected</u>:

by the remote service provider because the dialogue profile (see 4.3.3.1) in the request included the DIPE-MULTISYNCHRONIZE service which the remote service user does not support.

<u>DIPE dialogue rejected</u>:

by the remote service provider because the DIPE program name in the destination reference (see 4.1.3.1) is not recognized.

<u>DIPE dialogue rejected</u>:

by the remote service provider because the DIPE program name in the destination reference (see 4.1.3.1) is recognized, but the DIPE program is persistently not available.

<u>DIPE dialogue rejected</u>:

by the remote service provider because the DIPE program name in the destination reference (see 4.1.3.1) is recognized, but the DIPE program is temporarily unavailable.

<u>Remote program error</u>:

execution of the remote DIPE program instance has been prematurely terminated because the remote service provider (a) detected a program error in the execution of the remote service user, or (b) rejected an attempted incorrect use of DIPE services by the remote service user.

<u>DIPE service failure</u>:

The DIPE dialogue has been prematurely terminated because of some persistent failure within the DIPE service.

<u>DIPE service failure</u>:

The DIPE dialogue has been prematurely terminated because of some temporary failure within the DIPE service.

4.1.3.5 <u>Error types for termination of a DIPE dialogue by the service users</u>

It is proposed that the error type parameter identifies one of the following reasons for the service user terminating the dialogue abnormally. Further details may be recorded in error logs, see 4.1.3.7.

<u>Unrecoverable error</u>

The remote service user requested abnormal termination of the DIPE dialogue.

Unrecoverable error

The remote service user, who is also the provider of DIPE services, requested abnormal termination of the DIPE dialogue.

Unrecoverable error

The remote service user, who is also the provider of DIPE services, requested abnormal termination of the DIPE dialogue because of some time-out condition.

4.1.3.6 Error types for errors within a DIPE dialogue

It is proposed that the error type parameter identifies one of the following error notification reasons.

Program error

The remote service user issued the error notification when holding the token, but this did not result in truncation of an SDU.

Program error

The remote service user issued the error notification when holding the token, and this has resulted in truncation of an SDU.

Program error

The remote service user issued the error notification when not holding the token, and this may have caused information to be purged.

Program error

The remote service user, who is also the provider of DIPE services, issued the error notification when holding the token, but this did not result in truncation of an SDU.

Program error

The remote service user, who is also the provider of DIPE services, issued the error notification when holding the token, and this has resulted in truncation of an SDU.

Program error

The remote service user, who is also the provider of DIPE services, issued the error notification when not holding the token, and this may have caused information to be purged.

4.1.3.7 Error log information

Some abnormal termination and error events of the DIPE services may include information which could be inserted into error logs maintained by the DIPE service providers. The details are outside the scope of this TR.

See also DIPE Management services, section 4.5.

4.1.3.8 Service data unit parameter

The DIPE service data unit (SDU) is handled by the service as uninterpreted data. Size limits are for further study.

In the parameter tables of the service description, an X entry (which means that this parameter is present) includes the case of a null SDU, consisting of zero octets of data.

4.2 Unit-operation Service

4.2.1. Introduction

The DIPE Unit-operation service is designed to support OSI distributed transaction processing via execution of logically independent unitdata transfers and logically independent remote operations (e.g. those with the interaction structure defined in CCITT X.410).

There are four service elements:

- DIPE-UNITDATA service element (see 4.2.4.1),
- DIPE-OPERATION service element (see 4.2.4.2),
- DIPE-P-CANCEL service element (see 4.2.4.3),
- DIPE-U-CANCEL service element (see 4.2.4.4).

These services are connectionless in that there is no separate connection establishment phase; but they have some of the characteristics of connection-oriented working.

4.2.2 Common concepts

4.2.2.1 Dialogue-establishment information

A temporary DIPE dialogue exists for the duration of the execution of a DIPE-UNITDATA or DIPE-OPERATION service element. It may be terminated prematurely with an event of one of the cancellation services.

The parameters which define the characteristics of this temporary DIPE dialogue are grouped together in the service description, and are termed "dialogue establishment information".

4.2.2.2 Time-out

Because each use of the DIPE-OPERATION service element is necessarily a complete and confirmed interaction, some last resort means of abnormal termination is needed if expected events do not occur.

A spontaneous local time-out is provided for this purpose. It occurs as a DIPE-P-CANCEL indication event. These events occur locally and are not directly synchronized and do not necessarily occur at both ends (see 4.2.4.3).

The time-out values are derived from the confirmation time-out parameter (see 4.2.3.3).

4.2.2.3 Token

There is no explicit token in these unit-operation services. Alternating roles are implicit in the DIPE-OPERATION service. Also multiple unit-operation DIPE dialogues may occur concurrently, and in both directions, between the same DIPE service users.

4.2.3 Parameters

The parameters of the DIPE Unit-operation service elements are those defined below, and a selection of the common parameters defined in section 4.1.3.

4.2.3.1 Request identifier parameter

Unique reference, similar to CCITT X.410 ROS/ROP invoke identifier. Needs further study.

4.2.3.2 Maximum SDU size parameter

This value is provided to assist the service providers and the destination service user. It is an estimated size, and does not delimit the SDU. Its use requires further study.

4.2.3.3 Confirmation time-out parameter

This sets the time-out interval, the use of which is described in section 4.2.2.2. The range of time-out values is for further study.

4.2.3.4 Anticipation index parameter

This is an estimate of the probability of further unit-operation DIPE dialogues between the same service users soon. The service providers may use this information to optimize protocol interactions and scheduling. Details are for future study.

4.2.3.5 Quality of service parameter

For future study.

4.2.3.6 Priority classification parameter

This is a means to distinguish the relative priority of different unit-operation DIPE dialogues. Details are for future study.

4.2.3.7 Exception type parameter

For future study.

4.2.3.8 Service data unit (SDU)

See 4.1.3.8.

4.2.4 Service definition

4.2.4.1 DIPE-UNITDATA

This DIPE service element provides a DIPE dialogue which consists of a single connectionless data transfer from the requestor to the referenced destination. There is no provision for a reply from the destination or for confirmation of successful delivery.

The outcome is required to be that the indication event either occurs once, or does not occur at all (it should never occur more than once). The QOS parameter defines the required probability of the indication event occuring. Typically this is a high probability.

Parameters	R	I
Dialogue establishment information:		
- Request identifier (see 4.2.3.1)	X	=
- Destination reference (see 4.1.3.1)	X	-
- Source reference (see 4.1.3.2)	-	X
- Max. SDU size (see 4.2.3.2)	(X)	=
- Anticipation index (see 4.2.3.4)	(X)	=
- QOS (see 4.2.3.5)	(X)	-
- Authentication (see 4.1.3.3)	(X)	-
- Priority classification (see 4.2.3.6)	(X)	=
- Exception type (see 4.2.3.7)	X	=
SDU (see 4.2.3.8)	X	=

See also the event tables referenced in Appendix B.

4.2.4.2 DIPE-OPERATION

This DIPE service element provides a DIPE dialogue which consists of a data transfer from the requestor to the referenced destination, and a reply data transfer from the destination to the requestor.

There may also be cancellation events which disruptively terminate this service, see 4.2.4.3 and 4.2.4.4.

The outcome of the request is required to be that the indication event occurs once, or does not occur at all (it should never occur more than once). The same requirement applies to the occurence of the confirmation event after the response event. The QOS parameter defines the required probability of these indication and confirmation events occuring. Typically this is a high probability.

Occurence of the confirmation event guarantees that the destination service user has received the indication event. It is required that if the confirmation event does not occur within the time-out interval, a DIPE-P-CANCEL indication event will occur instead.

	R	I	R	C
Dialogue establishment information:				
- Request identifier (see 4.2.3.1)	X	=	=	=
- Destination reference (see 4.1.3.1)	X	-	-	-
- Source reference (see 4.1.3.2)	-	X	-	-
- Max. SDU size for req. (see 4.2.3.2)	(X)	=	-	-
- Max. SDU size for resp. (see 4.2.3.2)	(X)	=	(X)	=
- Confirmation time-out (see 4.2.3.3)	(X)	=	-	-
- Anticipation index (see 4.2.3.4)	(X)	=	(X)	-
- QOS (see 4.2.3.5)	(X)	-	=	-
- Authentication (see 4.1.3.3)	(X)	-	-	-
- Priority classification (see 4.2.3.6)	(X)	=	-	-
- Exception type (see 4.2.3.7)	X	=	X	=
SDU (see 4.2.3.8)	X	=	X	=

See also the event tables referenced in Appendix B.

4.2.4.3 DIPE-P-CANCEL

This DIPE service element provides the means for the DIPE service providers to unconditionally cancel a unit-operation DIPE dialogue.

The results may include truncation of DIPE-OPERATION service events, which then carry the corresponding exception code. The DIPE-P-CANCEL indication event may occur at both ends of the DIPE dialogue, or only one (if the other end has already terminated).

Parameter	I	I
Request identifier (see 4.2.3.1)	X	=
Exception type (see below)	X	=

The exception type is any of the DIPE dialogue service provider termination types defined in 4.2.3.7.

Use of this service may result in error log information (see 4.1.3.7).

4.2.4.4 DIPE-U-CANCEL

This DIPE service element provides the means for the requestor of a unit-operation DIPE dialogue to unconditionally cancel it.

The requestor of this service must be the requestor of the DIPE-operation which is referenced by the request identifier. The results may include truncation of DIPE-OPERATION service indication, response and confirmation events. They then reference the corresponding exception type.

The outcome of the request is required to be that the indication event (or an equivalent DIPE-OPERATION indication or response exception event) either occurs once, or does not occur at all (it should never occur more than once). The QOS parameter in the DIPE-OPERATION request defines the required probability of the indication event occuring. Typically this is a high probability.

	R	I
Control information:		
- Request identifier (see 4.2.3.1)	X	=
- Anticipation index (see 4.2.3.4)	X	=
- Exception type (see below)	X	=

The exception types are any of the service user termination codes defined in 4.2.3.7.

See also the event tables referenced in Appendix B.

The other service user may achieve a termination equivalent to DIPE-U-CANCEL by means of the exception type in his DIPE-OPERATION response event.

Use of this service element may result in error log information (see 4.1.3.7).

4.3 Connection-oriented Service

4.3.1 Introduction

The DIPE connection-oriented service is designed to support OSI distributed transaction processing through a series of service elements whose execution is logically inter-dependent.

These services are:

- DIPE-BEGIN Service element
- DIPE-P-ABORT Service element
- DIPE-U-ABORT Service element
- DIPE-END Service element
- DIPE-DATA Service element
- DIPE-GIVE Service element
- DIPE-PLEASE Service element
- DIPE-HANDSHAKE Service element
- DIPE-ERROR Service element

The multisynchronization services defined in section 4.4 are extensions to these connection-oriented services.

4.3.2 Common concepts

4.3.2.1 Performance

These services are designed to allow performance optimizations to satisfy requirement Q1 in section 2.4.

The main characteristic is that most of these DIPE service elements have the unconfirmed structure (request and indication only). This gives maximum possibility to concatenate their protocol effects, including blocking together of SDUs. (The related buffering optimization in implementations would probably need to include a local "push" element, which cannot be visible in an OSI service definition.).

4.3.2.2 Token

The connection-oriented services have a two-way alternate dialogue structure (TWA). A single token is used to control all alternating characteristics of this dialogue. Certain services may be requested only by the current holder of the token.

4.3.2.3 Combination of service elements

DIPE connection-oriented service elements with unconfirmed interaction structure (R.I), may be combined with an associated confirmed service element (R.I.R.C.). This allows richer services to be constructed, and provides opportunities for optimizations. See section 4.3.4.4, 4.3.4.6, 4.3.4.9, 4.4.3 and 4.4.4.

4.3.3 Parameters

The parameters of the DIPE connection-oriented service elements are those defined below, and a selection of the common parameters defined in section 4.1.3.

4.3.3.1 Dialogue profile parameter

A DIPE dialogue profile is the specification of what DIPE services are to be used, and how they are to be used. The details of its content are for further study; but it would probably include selection of options (see 4.6) and size restrictions.

4.3.3.2 Quality-of-service parameter

For future study.

4.3.3.3 Initialization parameter

The DIPE-BEGIN service includes optional "initialization parameters". Their values are transferred by the service as uninterpreted data. This feature is provided to enable automatic initialization of program data variables. Size limits are for further study.

4.3.3.4 Error types of the DIPE-P-ABORT service

All the error types defined in section 4.1.3.4 are likely to be supported.

4.3.3.5 Error types of the DIPE-U-ABORT service

All the error types defined in section 4.1.3.5 are likely to be supported.

4.3.3.6 Error types of the DIPE-ERROR service

All the error types defined in section 4.1.3.6 are likely to be supported.

4.3.4 Service definition

4.3.4.1 DIPE-BEGIN

This DIPE service element is used to begin a connection-oriented DIPE dialogue between two DIPE program instances.

Parameters	R	I
Destination reference (see 4.1.3.1)	X	-
Source reference (see 4.1.3.2)	-	X
Dialogue profile (see 4.3.3.1)	X	=
Quality-of-service (see 4.3.3.2)	(X)	-
Authentication reference (see 4.1.3.3)	(X)	-
Initialization parameters (see 4.3.3.3)	(X)	-

Because there is no confirmation event to wait for after the request event, the requestor is allowed to continue the dialogue and issue other requests (e.g. a DIPE-DATA request, a DIPE-GIVE request, a DIPE-HANDSHAKE request, etc.). These other requests might be used to construct a complete round trip handshake to exchange user-defined control information before further interactions.

The unconfirmed service structure of DIPE-BEGIN provides an opportunity for the DIPE service providers to optimize transfers across the network (e.g. by concatenation) and to optimize scheduling of program execution.

The indication event will only occur if the DIPE-BEGIN is successful. If the remote service provider is unable or unwilling to accept the request, it rejects it (there is no negotiation). This is done by causing a DIPE-P-ABORT indication to the requestor (with the appropriate reason code). This abnormal termination prevents the intended DIPE dialogue, and there are no indication events of any kind to the destination service user (any other requests already issued have no effect at the remote service user). A DIPE-U-ABORT or DIPE-P-ABORT request initiated for other reasons in similar circumstances may also have this no-indication effect.

There is no provision for a rejection directly by the destination DIPE service user. This effect can be achieved by him subsequently using the DIPE-U-ABORT service.

After the DIPE-BEGIN service request, the requestor is the current holder of the token.

4.3.4.2 DIPE-P-ABORT

This DIPE service element is used by the DIPE service providers for unconditional termination of a connection-oriented DIPE dialogue.

Parameters	I	I
Error type (see 4.3.3.4)	X	X

The service is destructive: events in transit or queued at their destination are purged.

Use of this service element may result in error-log information (see 4.1.3.7).

4.3.4.3 DIPE-U-ABORT

This DIPE service element is used by DIPE service users for abnormal termination of a connection-oriented DIPE dialogue.

Parameters	R	I
Error type (see 4.3.3.5)	X	=

Abnormal termination is unconditional and has destructive effects. If the requestor is not the current holder of the token, events which are in transit from the other service user will be purged. If the requestor is the current holder of the token, any current data transfer (DIPE-DATA service) may be truncated. These effects need further study.

Use of this service element may result in error-log information (see 4.1.3.7).

4.3.4.4 DIPE-END

This DIPE service element is used by DIPE service users for normal termination of a connection-oriented DIPE dialogue. The requestor of the service is required to have the token.

Parameters	R	I
Associated request (see 4.3.2.3)	(X)	=

Normal termination is generally non-destructive. Services previously requested by the requestor are completed normally. But any events which are in transit from the other service user are discarded (these can only be the disruptive indication events DIPE-ERROR, or DIPE-P-ABORT, or DIPE-U-ABORT).

Extra synchronization, which avoids the above effect, can be achieved by a combination (see 4.3.2.3) which makes completion of the DIPE-END request conditional on normal completion of a DIPE-HANDSHAKE request or a DIPE-MULTISYNCHRONIZE request. The requestor is held inactive and waiting until normal completion of the synchronization, after which the DIPE-END takes effect. If the synchronization does not complete normally, the DIPE-END is cancelled.

4.3.4.5 DIPE-DATA

This DIPE service element is used to transfer data between DIPE service users. The requestor of the service is required to have the token.

Parameters	R	I
SDU (see 4.1.3.8)	X	=

4.3.4.6 DIPE-GIVE

This DIPE service element is the normal means for DIPE service users to exchange the token. The requestor of the service is required to have the token.

Parameters	R	I
Associated request (see 4.3.2.3)	(X)	=

The service may be combined (see 4.3.2.3) with a DIPE-HANDSHAKE request or a DIPE-MULTISYNCHRONIZE request. In such cases, the token is not available to the recipient until after the associated service element has been completed locally.

4.3.4.7 DIPE-PLEASE

This DIPE service element is the means to request transfer of the token to the requestor.

It is an unconfirmed service. There are no parameters.

This is a notification service. The service user receiving the indication makes his own choice when to send the token (by making a DIPE-GIVE request), or else not to send it.

4.3.4.8 DIPE-ERROR

This DIPE service element provides error notification between DIPE service users, with related synchronization of the DIPE dialogue. The requestor is held inactive and waiting until after the corresponding indication event has occurred (or an intervening disruptive indication event).

Use of this service may disrupt the effect of other services, as described below.

Parameters	R	I
Error type (see 4.3.3.6)	-	X

If the request is made by the token holder, the service has no disruptive effects. The error type parameter value in the indication event distinguishes whether or not an SDU has been truncated.

If the request is made by the service user who does not hold the token, the service may have disruptive effects on elements in transit from the other service user. The error type parameter value in the indication event reports that data has been purged; but it does not distinguish whether or not there was actually anything to be purged. Further study is needed to clarify the details.

Cases of potential collisions between DIPE-ERROR service events are for further study.

Use of this service element may result in error-log information (see 4.1.3.7).

4.3.4.9 DIPE-HANDSHAKE

This DIPE service element provides a synchronizing handshake between the two users of a DIPE interaction. The requestor of the service is required to have the token.

This is a confirmed service. The events have no parameters.

By completing this handshake normally, the DIPE service users confirm that their processing is completed for all previous DIPE service events in this DIPE dialogue. The requestor is held inactive and waiting until his confirmation event or a disruptive indication event. The responder is held inactive between the indication and response events. This synchronizes the two service users.

The DIPE service user receiving the indication event may reject the handshake by making a DIPE-ERROR or DIPE-U-ABORT request.

This service element may also be combined with others (see 4.3.2.3). The combinations are with the DIPE-GIVE service element (see 4.3.4.6) and the DIPE-END service element (see 4.3.4.4).

4.4 Multisynchronization Service

4.4.1 Introduction

The DIPE Multisynchronization service is an extension to the DIPE connection-oriented service described in section 4.3 above. These two DIPE service structures and the optimizations within them are designed to be mutually supportive. The DIPE Multisynchronization service provides interlocking synchronization across a DIPE processing tree by enforcing that all the relevant DIPE programs issue a DIPE-MULTISYNCHRONIZE request.

This synchronization supports two-phase commitment and recovery (CCR), which is needed for the integrity of some applications. The relationship between this service and the current ISO/TC97/SC21 work on CCR requires further study.

4.4.2 DIPE-MULTISYNCHRONIZE service element

For each invocation of the DIPE-MULTISYNCHRONIZE service, the service interaction structure and parameters are shown below. The request is general to all the requestor's DIPE dialogues. He is required to be the current holder of the token for all his current DIPE dialogues, except any on which he has currently received a DIPE-MULTISYNCHRONIZE indication event.

Parameters	R	I		C
Direction (forwards or backwards)	X	X		X

The synchronization direction "forwards" means commitment (requested or confirmed); "backwards" means backout. By nominating a synchronization direction in the request event, the requestor is "voting" for it.

The request causes the indication event to occur at the other end of those of his DIPE dialogues which have a dialogue profile which includes use of this service (but not one on which he has currently received a DIPE-MULTI-SYNCHRONIZE indication event). The requestor is held inactive and waiting until his DIPE-MULTISYNCHRONIZE confirmation event. These request and confirmation events are therefore combined as a single indivisible event from the viewpoint of the requestor.

4.4.3 Explanation

This service is designed to be invariant to the "instantiation" direction of the DIPE processing tree, invariant to the positioning of the DIPE users in any commitment hierarchy, invariant to the number of active DIPE dialogues, invariant to whether these dialogues are single-endpoint or multi-endpoint, invariant to the total size of the DIPE processing tree, and invariant to whether or not all the DIPE dialogues in the DIPE processing tree include this service or use the other DIPE connection-oriented service, or use the DIPE Unit Operation service.

Use of the service determines for all its users the same outcome. This is either to synchronize forwards (i.e. commit) or to synchronize backwards (i.e. backout). Their actual processing actions (securing results by use of stable storage, or commitment, or rollback, etc.) are not visible to the DIPE service. The service users may include system-provided database managers etc., so these potentially complex processing actions of DIPE service users are not necessarily visible to the end user. The multisynchronization service is designed to be reliable in the presence of communication failures and DIPE node crashes and DIPE program crashes.

Each DIPE service user receiving an indication event of this service is required subsequently to issue a separate DIPE-MULTISYNCHRONIZE request. This causes DIPE-MULTISYNCHRONIZE indications at the other end of each of his active DIPE dialogues that include this service (but not one on which he has currently received a DIPE-MULTISYNCHRONIZE indication event). The nominated direction must be backwards if this was nominated in an indication event received; otherwise it can be backwards or forwards. One reason for nominating the backwards direction could be that the indication event violates a commitment hierarchy known to the DIPE service user.

These interlocking invocations of the DIPE-MULTISYNCHRONIZE service spread (from one or more origins) across all the DIPE dialogues linked in this way within a DIPE processing tree, until they are all inactive. The required token positioning for each service user affected also quiesces their other DIPE dialogues which do not include this service in their dialogue profile. For these DIPE dialogues the service users may use the DIPE-HANDSHAKE service in ways which have the effect of extending the commitment structure into outer branches of the DIPE processing tree. This is particularly useful where outer branches have simple client/server access interactions. The more complex commitment processing is then hidden and internal to the server(s) concerned.

Protocols inside the DIPE service control this synchronization and analyse the votes.

If the vote of all the relevant DIPE service users was for the forward direction, the service reliably provides to all of them, as soon as possible, a confirmation nominating the forwards direction (on which each should take whatever is their agreed action).

If the vote of any of the relevant DIPE service users was for the backwards direction, the service reliably provides to all of them, as soon as possible, a confirmation event nominating the backwards direction (on which each should take whatever is their agreed action).

Any intervening DIPE-ERROR or DIPE-U-ABORT or DIPE-P-ABORT service event which disrupts this voting, may result in the service provider(s) nominating the backwards direction. The service providers include sophisticated recovery management provisions to deal with cases where persistent failures occur during critical stages of the voting and confirmation procedures.

4.4.4 Combination with DIPE-END

Normal termination of a DIPE dialogue may be combined (see 4.3.2.3) with use of the DIPE-MULTISYNCHRONIZE service. The termination of the specified DIPE dialogue is made conditional on normal (forwards) completion of the DIPE-MULTISYNCHRONIZE service. For further details see the event tables referenced in Appendix B.

4.4.5 Combination with DIPE-GIVE

The giving of the token may be delayed and made conditional by combining it with use of the DIPE-MULTISYNCHRONIZE service (see 4.3.2.3). For further details see the event tables referenced in Appendix B.

4.5 DIPE Management Service

No specific service elements are proposed at this stage of the work. A general approach is outlined in section 3.6.5. This subject needs further study.

4.6 Option Sets

Some users of DIPE services would not need all of the facilities. Standard option sets are a way of avoiding incompatible choices.

For the DIPE connection-oriented service and the DIPE multisynchronization service, the option sets defined in reference IBM/1 are a possible basis.

For the DIPE Unit Operation service there are no specific option set proposals at this stage of the work.

This subject needs further study.

5. ITEMS FOR FURTHER STUDY

5.1 Classification of Items for Further Study

DIPE in OSI structure

- positioning in the OSI layering
- use of OSI naming and addressing

Outlines of protocol

- for DIPE unit operation
- for DIPE connection-oriented
- their relationship to current ISO connection-oriented and connectionless protocols

Encoding of protocol

- for DIPE unit operation
- for DIPE connection-oriented

Interface definition

- for DIPE unit operation
- for DIPE connection-oriented

Additional services and protocol

- DIPE multi-endpoint service
- DIPE management service
- quality of service

Authentication

SDU size limits

5.2 Cross-references to the Issues in the Current Text

2.3	note: Interface
3.6.1	positioning in OSI layering
3.6.2	completeness of DIPE service
3.6.4	multi-endpoint service
3.6.5	management service
3.7	user interface issues
4.1.1	completeness of DIPE service
4.1.3.1	size of the identifiers
4.1.3.3	security attributes
4.1.3.4	error type completeness
4.1.3.8	SDU size
4.2.3.1	request identifier parameter
4.2.3.2	maximum SDU size parameter
4.2.3.3	confirmation time-out parameter
4.2.3.4	anticipation-index parameter
4.2.3.5	quality of service parameter
4.2.3.6	priority classification parameter
4.2.3.7	exception type parameter
4.3.3.2	quality of service parameter

4.3.3.3	initialization parameter size limits
4.3.4.3	effect of U-ABORT
4.3.4.8	collision of DIPE-ERROR
4.4	Multisynchronization and CCR
4.5	management services
4.6	option sets

APPENDIX A

PERFORMANCE ANALYSIS METHODOLOGY AND TARGETS

A.1 Overview

This appendix discusses performance requirements, defines a simple methodology to analyse and compare the key performance efficiency characteristics of DIPE protocol standards, and sets some targets.

These targets are based on the assumption that the performance characteristics required are whatever is needed to achieve performance efficiency which is comparable with the best commercially available networking systems.

A.2 What Performance Needs ?

The performance requirements placed on the DIPE service vary with the characteristics of the applications using the DIPE service:

- Some interactive applications are most concerned with response time.
- Some interactive applications tend to be more concerned with efficient use of transmission resources.
- Some interactive applications are concerned with both response time and efficient transmission, and the relative importance of these varies with time and current work content.

The optimum solution may vary with transmission media characteristics:

- With high speed LAN, the number of round-trip-delays has relatively little impact on performance, because they are short.
- With a WAN, the number of round-trip-delays may be critically important, because they are relatively long (especially with packet-switching, or satellite links, or multi-hop communication).
- Communications technical characteristics and technical structures may be sensitive to the number of transmissions, or to the size of transmissions, or call duration, etc.

The optimum solution may vary with end-system characteristics:

- Some systems may benefit from minimizing execution path lengths; in others this may be relatively unimportant.
- Some systems may benefit from minimizing buffer size and occupancy; in others the opposite may be the case.

The standards providing the DIPE services should be capable of the different performance adaptations needed in such widely differing circumstances. This includes the dynamically changing requirements of the third group of applications in the above list. Therefore, flexibility to meet different performance needs is a key requirement.

A.3 Approach to the Analysis

The subject of the analysis is the protocol standards, not the implementations. The methodology should measure characteristics which can be observed by examining the protocol specifications in the standards documents.

The methodology should be consistent and its results quantitative, so that valid and objective comparisons and choices can be made.

The methodology should be simple to use, and should concentrate on a few key performance characteristics which will have the most impact on DIPE standards design decisions. Complete modelling of performance characteristics is not a current aim.

These criteria restrict the choice of measures. Those chosen are:

a) The number of transfers. This affects communications efficiency and processing efficiency in various ways, and is often the dominant contributor to processing overheads.

b) The number of round-trip delays. This affects response time.

c) The size of the specification needed to define the protocol completely. With current specification methods, the generally accessible measurement is the number of state/event/condition entries in the protocol state/event tables. This affects the complexity and size of implementation, and the difficulty of validation and maintenance.

d) Transfer size. This affects communications efficiency.

e) Variability of response time when there are error recoveries or abnormal timing circumstances. If excessive, this adversely affects the user's perception of response times.

Another important characteristic is the structure and complexity of data encoding and decoding, which affects processing costs. A measure for this is for future study. However, this characteristic is relatively unimportant for DIPE itself. But it is crucial for higher levels of protocol, e.g. the transfer syntax which is carried as uninterpreted data by DIPE.

Finally, the most important characteristic to be assessed is flexibility. As explained in A.2 above, the standards must enable implementations to adapt to varying circumstances, statically and dynamically.

A.4 Measurement Procedure

The first step is to select and specify some representative units of work to be used as test cases in evaluating performance characteristics.

It may be important to examine the variability of performance characteristics when there are error recoveries or abnormal timing circumstances. To explore these, extra test cases should be specified, and a relative probability should be assigned to all the test cases (expressed as decimal fractions summing to 1.0).

The result is an agreed schedule of test cases, against which the same measurements can be made for the different sets of protocol standards and alternative designs which are examined.

For a set of protocol standards to be evaluated, the following measurements are taken:

a) For each normal test case, count the total number of separate transfers between systems.

b) For each normal test case, count the total number of between-system round-trip delays which are inherent in the structure of the protocol(s).

c) Count the total number of transitions (state/event/condition combinations) in the state/event tables (of the conformance subsets) of the standards needed to support the complete schedule of test cases.

d) Count the total size, in octets, of the transfers in each normal test case.

e) If performance variability is to be measured for a unit of work:

   - For each exception test case, count the total number of between-system round-trip delays which are inherent in the structure of the transfers (also making equivalent allowances for any protocol timer effects).

   - Using the probabilities defined in the test schedule, together with the above exception counts and the corresponding normal counts b), calculate the probability distribution of the count values.

The points at which all the measurements are to be made are:

- network-service-data-units (NSDUs) for measurements a), b), d) and e), assuming unlimited NSDU size;

- for measurements c) the state/event tables of all the protocols used below the DIPE service and above the network layer service.

The measurement method is examination and analysis of the specifications in the set of standards involved. There may be various combinations of standards (e.g. when different protocols provide equivalent services), and various possible outcomes depending on legitimate implementation and operational choices.

A.5 Performance Targets

The aim is that standards should be capable of providing minimum (i.e. excellent) values of all the measures, but subject to the following criteria.

1) The values (a), (b), (d) and (e) shall be free to vary in ways dependent on implementation choices and operational choices which are outside the scope of the standards (e.g. buffer sizes, actual duration of round-trip delays, whether the actual usage is of a type which benefits from concatenation, etc.).
2) The achievable values of both (a) and (b) should be as close as possible to the minima theoretically derivable from high-level logical analysis of the units of work concerned.
3) The values (a) and (b) are of primary importance. The other values (c), (d) and (e) are of secondary importance, and may be higher (worse) if this enables lower (better) values of (a) and/or (b).
4) The performance variability (e) should be such that there is, say, a 95% probability of the delay count value (b) not exceeding 1,5 times the mean.

The rationale is explained in A.6 below.

A.6 Rationale

The choice of what to measure has already been justified in A.3 above.

The rationale of the targets 1) to 4) set in A.5 above is as follows:

1) This assures the flexibility to allow adaptation to different circumstances.
2) This maximizes the opportunities to optimize processing overheads, transmission efficiency and response times. It is also the level of efficiency achievable by leading proprietary protocols.
3) This concentrates attention on the overall effects of communication in a networked system.
4) This helps usability.

The measurements are carried down through the layers because the aim is to optimize the net performance of all the layers when they are working together. Some intermediate analysis can, of course, be done on individual layers, and without looking behind layer service boundaries.

The measurements stop at the network service because this is the point at which there are strong interactions with variable characteristics which are network-specific. Measuring and optimizing the interactions within the lower layer is very important, but can be considered separately, and can be integrated subsequently with upper layer measurements.

APPENDIX B

FORMAL SPECIFICATION

Section 4 of this Technical Report provides a natural language specification of the DIPE service. The work done to produce this included use of detailed state-event tables and event sequence tables. These are carried forward in ECMA working papers. Current copies of ECMA working papers are available from the ECMA Secretariat.