Fundamentals of the Layer Service Definitions and Protocol Specifications

PETER F. LININGTON

Invited Paper

Abstract—The paper discusses the modeling techniques used in Open Systems Interconnection standardization. There is a need for sudivision of the standardization work in order to allow orderly management of development, and the means of achieving this modularity by use of the concepts of service and protocol is explained. The conventions used in defining services are set out, so as to form an introduction to the detailed service descriptions in later papers. Various extensions to the basic concepts of service and protocol are presented, and the relation of the models established to the real world are discussed.

I. Introduction

An EARLIER PAPER in this issue introduced the concept of Open Systems Interconnection (OSI) and its supporting Reference Model. This paper presents some of the specification techniques used in the work, concentrating principally on those concerned with the standardization of Services and Protocols. The concepts introduced here will be drawn on in the later papers which deal with specific aspects of the work.

The aim of OSI is to remove any technical impediment from communication between systems, even though they may be of quite different origins. OSI is not concerned with the description of the internal operation of any single system; it is concerned only with the exchange of information at the points of interconnection between systems.

The description of the communication is organized by applying various techniques to produce a modular structure. The major subdivisions, within which all other OSI standards are to be interpreted, are set out in the Reference Model [1].

The advantages of a modular approach are well known in many areas of human activity. Few significant enterprises are simple enough to be understood as a whole by a single individual. A modular approach provides a comprehensible structure, and subdivides the work into individual pieces of manageable size. These can then be subject to independent development and maintenance. This last is particularly important in establishing standards appropriate to a rapidly developing technology, where innovation will require addition and updating of the individual parts to be performed without destabilizing the whole set.

The techniques are applied to form an interlocking family of standards which make up a closed formal system. Taken as a whole, they are mutually consistent and complete, in that they contain all necessary definitions and constraints. However, the use of these standards goes beyond this formal system in setting up elements in the real world which conform to the standards. The relation of standards to the products which conform to them is discussed briefly at the end of this paper.

Manuscript received August 31, 1983.

The author is with Rutherford Appleton Laboratory, Chilton, Didcot, Oxon OX11 0QX, England.

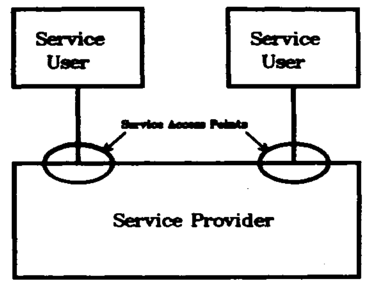

Fig. 1. The Service Model.

II. The Concept of Service

Before two systems can communicate, they must share a common set of rules for generating and interpreting the messages they send or receive. This set of rules is complex and incorporates many different design decisions. A structured approach to defining and monitoring these rules is needed to allow the activity to be divided into manageable tasks, and to separate independent elements of technical choice. The Open Systems Interconnection Reference Model is an example of such a structured approach. It divides the interconnection protocols in a particular way by defining a series of layers of function. Each boundary in the model represents a demarkation between groups of functions; these functions are given visibility by different aspects of the protocol rules.

When the term "service" is used in the OSI work, it represents the definition of such a demarkation, and represents a boundary between functions. Protocols are normally viewed as a vertical sequence, with the most abstract statements of objectives at the top and successively more specific, technology-dependent constraints controlling functions which form a column supporting them.

In this sense, the service provides to those parts of a system above the service boundary, which are collectively called the user of the service (see Fig. 1), a statement of the set of capabilities offered, independent of the detailed way in which they are realized. To the parts of the system below the boundary, which are collectively called the provider of the service, it represents a set of constraints on the design process; any solution which satisfies these constraints will provide an equivalent set of capabilities to the user of the service.

The provider of the service will, in general, be distributed

0018-9219/83/1200-1341$01.00

across a number of physically distinct pieces of equipment. However, the way in which this subdivision takes place is not of concern to the service user. The user may view the service provider as a distributed abstract machine [7], the operation of which provides the specified service.

In the current basic set of OSI standards, most of the services defined involve one service provider and two users which engage in a dialogue. However, the Reference Model introduces other communication concepts, the development of which is planned within ISO. Two of these are the multi-endpoint connection and broadcast (or multicast) communication; they are both characterized by the transmission of messages from one originator to a number of recipients, and their description will involve services with more than two users.

The definition of a service establishes the properties of a type of communication. It establishes the relationship, in general, between any entity playing the role of user of the service, the other user, and the provider. However, any particular communication will be an example of this ideal relationship, and so is an instance of the type defined by the service. Commmunicating systems may be involved at any time in a number of such instances, which may be related parts of some larger activity, or may be independent.

The point of interaction between a service user and the service provider is called a service access point (see Fig. 1). When creating an instance of a communication service, the originating user informs the service provider of the identity of the service access point which gives access to the called user by quoting its address. Thus each service is associated, at least in principle, with an address space, within which the service's access points are identified. Each instance of a connection-oriented service associated with a particular service access point has a separate identity, visible as a connection endpoint, and identifiable within the local system by a connection endpoint identifier.

III. Layering

Use of the concept of service introduced above divides an activity into three parts, two of which communicate via the third. However, many subdivisions are required to divide a complex system into manageable parts, and so structuring techniques must be applied many times. It would, in principle, be possible to apply the service definition technique arbitrarily, so that the roles of user and provider in different instances overlapped in complex ways, but the interest in interconnection of pairs of systems leads naturally to a more regular application of the technique. Each of the service users represents a collection of functions within one of the systems to be interconnected, and the service provider represents the system functions in the two systems managing some aspect of the communication between them.

If the subdivisions generated are purely hierarchical, in that the service provider in one service instance comprises the whole of the service provider and part of each of the service users in another instance, the resulting structure is layered. A series of layers can be generated, each adding to the function of some "lower" service in order to provide some "higher" service. The rules which coordinate the functions thus isolated within the two communicating systems then form a protocol between the systems.

By convention, the layers are viewed as a vertical sequence. The number of functions within the service provider increases, and the level of abstraction of the service the user sees increases correspondingly as one progresses up the series of layers. It is conventional to number the layers from the bottom in ascending order, such that the layer numbered "N" uses the service numbered "$N-1$" and provides the service numbered "N" (see Fig. 2).

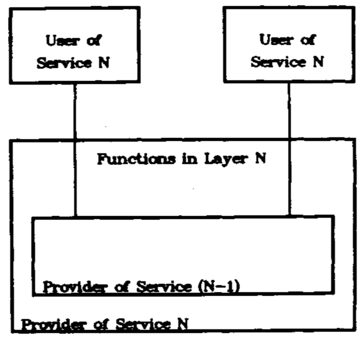

Fig. 2. The numbering of layers.

The user of service number "N" works entirely in terms of that service, and is not even aware, formally, that the division represented by any lower service exists. Similarly, the definition of service "N" does not imply or assume any particular higher layer subdivision of its users.

The choice of a division into a specific set of layered services is not unique. The particular choice of the seven layers used for OSI was described in a previous paper; it represents an engineering decision based on the balance of many conflicting factors. Other choices could have been made, but the adoption of a single universal choice leads to the reduction of variety which is the aim of all standardization.

IV. Protocols

Provision of a distributed communication service implies cooperation between the participating systems, and the rules which govern this cooperation constitute a protocol. Any protocol states procedural or syntactic constraints on the use of a lower layer service, with the objective of providing some higher layer service. Thus a syntactic rule might define a particular octet in any unit of data sent via the supporting service as representing a sequence number. An associated set of procedural rules might then define how this sequence number must be used to overcome transmission errors in the supporting service, the implication being a lower probability of error and a higher worst case transit delay in the supported service.

The majority of the protocol rules are often concerned with the use of a supporting data transfer service, but a complete protocol must specify an interpretation of all possible events in the supporting service, including such actions as connection establishment and error reporting where appropriate.

It is, in principle, a matter of formal proof to verify that the defined properties of the higher service can be derived from the protocol rules and the defined properties of the lower service. This process of proof is, however, not unique. Just as there can be an arbitrary number of equivalent proofs of a geometric theorem, there may be many different protocols which provide the same higher level of service. The Protocol Model for the OSI is presented in Fig. 3.

One of the main objectives of the layering technique was stated above to be the ability to change the detailed way a particular layer operates without affecting the whole structure. In particular, the objective is to be able to use alternative protocols to achieve the same service.

From the point of view of maximizing the probability of interconnection, there should clearly be only one single protocol

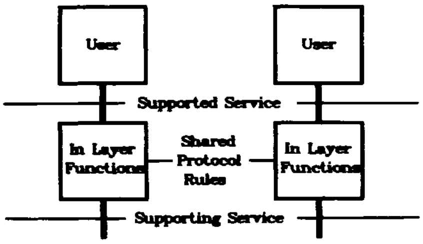

Fig. 3. The Protocol Model.

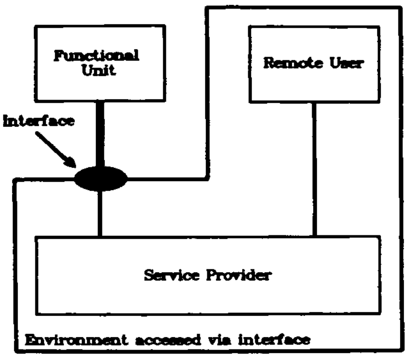

Fig. 4. The Interface Model.

for each layer, since systems will only communicate successfully if they are using the same protocol. However, other conflicting factors, such as different expected traffic patterns or balances of implementation cost have lead to the specification of alternative protocols in some layers of the Reference Model. The equivalence of these alternative protocols, in providing a single service, allows cost optimization within a layer without changing the environment within which other protocol layers operate.

V. Interfaces

Protocols are not the only aspect of system operation constrained by the service to be provided. Within a system, the boundaries between components form interfaces which can also contribute to modularity. OSI standardization is not concerned with interfaces; the standards only constrain communication between systems. However, there are other areas of standardization, such as the definition of programming or control languages, which are directly concerned with interfaces, and so their relation to the protocol standards is of considerable interest (see Fig. 4).

An interface is a boundary, or part of the boundary, between a component and its environment. The component operates on and is stimulated by its environment in terms of events at the interface. If the result of the interface interactions is some communication between a local part of the environment and parts in other systems, a protocol will be involved. There will be a correspondence between the operation of the interface and the resultant operations in the protocol. Since the service gives, in an abstract way, the capabilities of the distributed machine resulting from the operation of the protocol, it also relates to the operation of the interface. The interface gives access to mechanisms whose effects can be described by a combination of the definition of the communication service and the behavior of the remote user.

Just as there are many protocols which can provide the same service, so there can be many interfaces which can generate the same pattern of communication activity between systems. The choice of a particular interface style is system dependent and is left to the implementor of the OSI standards.

VI. Conventions Used in Services and Protocols

In order to provide a common specification style, a single set of descriptive conventions has been established, and is referenced by the individual service and protocol standards [2]. This document is itself to become a standard, and its text is currently being finalized within the ISO. The service conventions document aims to provide convergence of the natural language description of the standards; activity aimed at the production of formal description techniques to allow completely unambiguous standards is described in papers later in this issue.

The service conventions define the model of a service outlined in Section II, together with the roles of service user and service provider. The standard introduces the concept of a service primitive as an abstract, implementation-independent element of the interaction between the service user and the service provider. Four subtypes of service primitive are defined, corresponding to the major stages in an exchange between the service users. However, not all service elements involve all the types of primitive. The types are:

- request — a primitive issued by a service user to invoke some procedure;
- indication — a primitive issued by a service provider either to invoke some procedure or to indicate that a procedure has been invoked by the service user at the peer service access point;
- response — a primitive issued by a service user to complete, at a particular service access point, some procedure previously invoked by an indication at that service access point;
- confirm — a primitive issued by a service provider to complete, at a particular service access point, some procedure previously invoked by a request at that service access point.

Certain elements of dialogue occur repeatedly in the service definitions. The two commonest types are called:

- unconfirmed — a service element in which a request at one service access point leads only to an indication at the other;
- confirmed — a service element in which a request at one service access point leads to an indication at the other, which provokes the service user into issuing a response, leading to a confirm at the originating service access point.

The concept of primitive is introduced to allow description in terms of indivisible atomic events. Although the assumption of atomicity simplifies description of procedures, it arises from the choice of a particular level of abstraction at which to regard the service. Thus for example, data transfer is often regarded as an indivisible operation, in the definition of procedures, even though transmission of large amounts of data may take a considerable time, and there may be flow control within the transmission which operates on pieces of data smaller than the whole. The complete definition of the service may, therefore, involve statements describing different granularities to bring out different properties of the service. For example, an event which is seen as indivisible from the point of view of definition of valid sequences may need to be considered as series of smaller elements in order to describe some possible errors.

Subject to these considerations, however, the occurrence of a service primitive is a logically instantaneous and indivisible event.

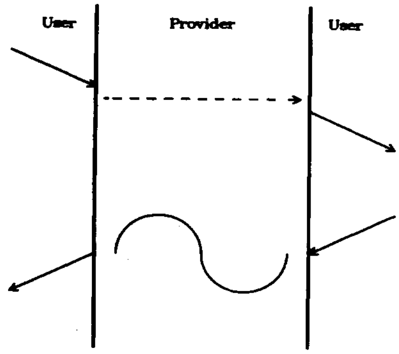

Fig. 5. Definition of time sequence.

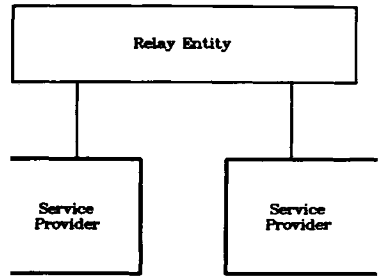

Fig. 6. The Relay Model.

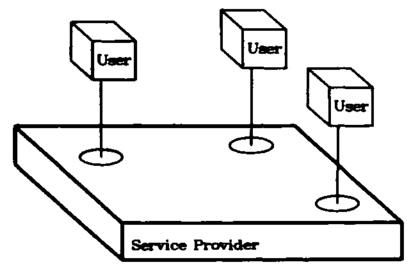

Fig. 7. A service with multiple users.

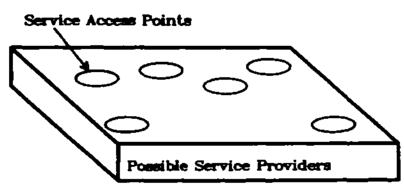

Fig. 8. The address space of a service.

The event occurs at a logically separate instant, which cannot be interrupted by another event. Each primitive has a particular direction, which is either from the service user to the service provider, or from the service provider to the service user. One or more parameters may be associated with a service primitive and each of these parameters has a defined range of values. Parameter values associated with a service primitive are passed in the direction of the service primitive.

The sequence in which related service primitives can occur is illustrated conventionally in a particular kind of time sequence diagram (see Fig. 5).

Each diagram is partitioned by two vertical lines into three fields. The central field represents the service provider and the two side fields represent the two service users. The lines represent passage of time (increasing downwards) at the two service access points between the users and the provider. Events are positioned along these lines to indicate their relative sequence. Arrows, placed in the areas representing the service users, indicate the direction of propagation of primitives, to or from the service provider; the angle of the arrow in the side field has no significance. Necessary sequence relations between the two service access points are emphasized by a horizontal dashed arrow between the time lines. Such sequencing may arise from a need for synchronization or from a causal relationship. In the absence of an arrow, no specific sequence is implied between points in time on the two lines. Where the layout of the diagram might otherwise suggest a spurious sequence, a tilde (~) is inserted between the lines to emphasize asynchrony.

VII. Types of Model

So far in this paper, the emphasis has been on the simple service with two users and on the protocols to support such a service. The elements are related by specific models, which were illustrated earlier in Sections II and IV.

These types of model can represent services or protocols supporting either connection-oriented or connectionless operation. They are not, however, sufficiently powerful to express all aspects of the OSI standardization, and certain extensions are required. The first area not covered by the simple concepts of service and protocol alone is the combination of media, which requires the introduction of a new kind of entity called a relay entity. This allows the correlation in a standard way of activity at two points of interconnection, one on each side of the relay entity. The main application of the technique is to the Network Layer of the Reference Model, where the processes of routing and of combining communication resources are described. Communication across more than one underlying service involves a new kind of model, in which one entity acts as a user of the two services in such a way that a composite communication service is created. The specification of a relay entity is, in effect, the specification of the interrelation of events in the two services. Since the relay provider links two service providers, it is effectively the dual of the simple service provider linking two users (see Fig. 6).

Another fairly immediate extension is the generalization of the number of service users. One such extension to handle broadcast communication was noted above, but there is another, in which each user is able to identify a number of peers individually. Some applications sublayers, such as that concerned with Job Transfer and Manipulation, and many management standards, define the relation of actions involving a larger number of service users (see Fig. 7). The distribution of results to a number of destinations, for example, implies a service model in which requests reference a number of peer users.

The above generalization is, however, still concerned with a single instance of communication, as for example in the description of a particular group of interacting management entities which make references to each other. A different kind of extension is required in the standardization of addressing schemes, and this is the first of two concerned with the relation of different instances of communication. The definition of an address space is concerned with the naming of the service access points between which communication instances could be established (see Fig. 8). If communication is established between a particular pair of service access points by quoting an address, an instance of the particular service is created, and as always an instance of an object has the general properties which characterize the type to which it belongs—in this case, the service type. If the service is connection-oriented, the communicating partners are bound together in a relationship which obviates the need to make explicit the distinction from all other possible communication instances at the same or any other service access points.

The final extension is concerned with the modeling of the relation of different communication instances. This has been introduced into the OSI work informally to describe some aspects of quality of service, and more precisely to capture the detailed properties of connectionless network services. The ability to make statements about the relation of two communication instances allows, for example, the expression of requirements for relative priority between two connections, or for the preservation of sequence between instances in an otherwise purely connectionless operation. It also allows the expression of technological constraints such as the inability to establish more than one connection at a time through a service access point which is supported by a circuit switched technology.

Other work is in progress which extends the modeling techniques still further, so as to provide a reference architecture for areas of standardization beyond interconnection. This work draws on techniques previously used for databases [3].

VIII. Implementation of Standards

The consistency relationship between a protocol and the service it supports was introduced above. When all the layers of protocol are considered together in the context of the Reference Model, the complete set of standards should form a closed formal system, in that it should consist of mutually consistent statements of the interrelation of a series of uninterpreted terms such as "entity," "service access point," or "physical medium." In the Application Layer, there will also be references to terms not related to communication, such as "file" or "unit of account" [5], [6]. Of course, the standards also contain strong informal indications of the intended use of the terms they relate in the real world, often implicit in the form of the names chosen.

However, while consistency between elements of the standards can be demonstrated, in principle, by verification techniques, there is no comparable procedure for testing the mapping of terms used in the standards onto the real world, without some hints from the implementor. Many different mappings are possible [4]. A vital step in the process is the assertion by some designer of the system under consideration that a particular component *is* a transport entity, or that a particular collection of equipment *is* an open system.

Once these assertions have been made, however, it is possible to observe behavior in the physical world and check that the various terms are assigned in a way which corresponds to the constraints given in the standards. Observation of the point asserted to be the physical medium can confirm the passage of signals which are interpretable as transport protocol exchanges when the component asserted to be a transport entity is stimulated. Similarly, the effects of insertion of signals at the alleged physical medium at other points asserted to have a particular role in the standards can be observed and checked for consistency with the relation of corresponding terms in the formal system.

This is the essence of the process of conformance testing. It depends on checking that an assertion by the constructor of some object about its correspondence with the formal system in the standards is consistent with the object's observed behavior.

IX. Conclusion

The paper has described briefly various of the modeling techniques used in the Open Systems Interconnection (OSI) standardization effort to express service definitions and protocol specifications. The application of these techniques allows the structuring of the standards work and thus provides a valuable tool for the management of a complex and many-faceted project. The definition of stable services allows the individual protocol standards to respond to continued technological progress without generating unforeseen side effects elsewhere in the structure.

Further papers in this issue use these techniques in discussing the specific content of the various standards for OSI.

References

[1] "Information processing systems—Open Systems Interconnection—Basic reference model," International Standards Organization, ISO 7498, 1982.

[2] "Information processing systems—Open Systems Interconnection—Service conventions," Working draft ISO/TC97/SC16 N1646.

[3] C. W. Bachman and R. G. Ross, "Towards a more complete Reference Model of computer based information systems," *Comput. Stand.*, vol. 1, p. 1, 1982.

[4] H. J. Burkhardt and T. H. Krecioch, "Standards for the ISO—Reference Model for Open Systems Interconnection," *Microproc. Microcomput.*, vol. 9, p. 237, 1982.

[5] P. F. Linington, "The upper layers of the ISO Reference Model," in *Proc ICCC*, M. B. Williams, Ed., p. 849, 1982.

[6] P. F. Linington, "The Virtual Filestore concept," *Comput. Networks*, to be published.

[7] S. Schindler, "Distributed abstract machine," *Comput. Commun.*, vol. 3, no. 5, p. 208, 1980.