T U M

I N S T I T U T F Ü R I N F O R M A T I K

How to Improve the Service Specifications of the ISO/OSI Basic Reference Model

Christian Facchi

TUM-I9615
März 1996

T E C H N I S C H E U N I V E R S I T Ä T M Ü N C H E N

TUM-INFO-3-1996-I9615-300/1.-FI

Typescript: ---

Druck: Mathematisches Institut und
Institut für Informatik der
Technischen Universität München

How to Improve the Service Specifications of the ISO/OSI Basic Reference Model

Christian Facchi*
Institut für Informatik
Technische Universität München
D-80290 München
E-Mail: facchi@informatik.tu-muenchen.de

Draft of March 21, 1996

Abstract

We present in an informal way the results of a formal specification of the ISO/OSI basic reference models service specifications. A new modular way of service specification which improves the reusability regarding different layers within a network is introduced. Some description techniques, which the ISO/OSI basic reference model uses, like time sequence diagrams, are clarified and improved. As a consequence the description of the ISO/OSI basic reference model gains a higher preciseness. Some ambiguities of the informal descriptions are removed by textual corrections.

1 Introduction

The OSI basic reference model describes different layers for the modularization of computer communication in [ISO84a, CCI88a]. According to [VL86] a specification of one layer should be based on the highest level of abstraction. Therefore we take service specifications instead of protocol specifications as requirement specifications, because a protocol describes only the implementation of a service.

*New address: Siemens AG; PN KE TCP 31; D-81359 München; Christian.Facchi@pn.siemens.de

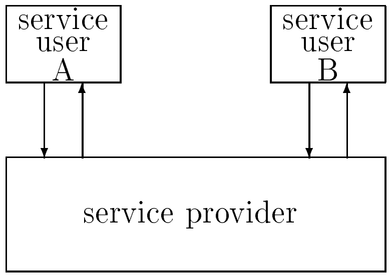

Figure 1: Abstract representation of one layer

Figure 1 shows an abstract representation of an arbitrary layer within the ISO/OSI basic reference model. The service users can exchange the basic elements of a communication, the so-called *service primitives*, only with the use of the service provider. So for a communication between the service users they have to take use of the service provider. The interfaces between each service user and the service provider are the *service access points (SAP)*. A main focus of our approach is to introduce a method describing the service provider of each layer in a formal way. For every layer there is a service specification as an ISO or CCITT[1] document. These specifications are based on natural language with some additions of semi-formal and formal descriptions. The natural language and the semi-formal parts may lead to not intended interpretation ambiguities, which should be avoided in an international standard. To avoid misinterpretation it is necessary to develop a formal service specification for every layer [Vis88]. Such a specification exists for the session and the transport layer [ISO92, ISO89b] using the formal description technique LOTOS [ISO89a]. However, due to the use of a monolithic specification style, according to [VSvSB91], the reusability of these specifications is not very high. Therefore in [Fac95b] a modular approach of a formal service specification has been introduced. This approach uses a constraint-oriented specification style [VSvSB91] that improves the reuseability of the specifications. In the following we present some improvements of the ISO documents or CCITT recommendations, which are based on the formal approach of the service specification.

Our method can be sketched as follows: As a first step the semi-formal parts of a service specification are transformed to formal ones in a schematic way. Then the natural language parts which are usually specific to one layer are formally described. The different specification parts are then combined into a complete and formal service specification.

In Section 2 we present a modular way for a formal service specification. Section 3 describes the results of the formalization of the description techniques used by the ISO and CCITT documents. Finally in Section 4 we give a summary.

[1]In 1993 the CCITT became the Telecommunication Standards Sector of the International Telecommunication Union (ITU-T). If a document is published by CCITT this organization name is used instead of ITU-T in the sequel.

2 The Modularization of Service Specifications

In the ISO[2] documents different description techniques are used. Each of them describes only one specific aspect of a service property. However, the combination of these different specification parts is not defined in the ISO documents. The presented modular approach is based on the service definition of the transport layer [ISO84b, CCI88f] which also can be seen in the definition of other layers. In this part one specific property has been defined:

> This section defines the constraints on the sequence in which the TS primitives may occur. The constraints determine the order in which TS primitives occur, but do not fully specify when they may occur. Other constraints, such as flow control of data, will affect the ability of a TS user or TS provider to issue a TS primitive at any particular time.

This definition suggests a modular approach to a service specification, in which for every part of the specification a separate description technique is used independently. To give a formal semantics we believe it is useful to define the semantics of every description technique and furthermore the semantics of the combination of them. Then each description part which is a specification of only one aspect of a service, can be regarded as an independent part. Because of the precise definition of the combination, these description parts can then be put together to describe a complete service. If this combination is defined as a conjunction, the specification style is *constraint-oriented* according to [VSvSB91]. However, such a precise interpretation is not carried out in the ISO documents in which only the parts of the specification have been mixed together. In [Fac95b] for every specification part an isolated formal semantics is developed and a operation for the combination is formally defined. This exact definition has among others the advantage of a higher reusability of the specifications for different layers. For each description technique in the ISO documents a method for developing a formal description has been introduced. Because the combination's formal definition major parts of the service specification can then be schematicly formalized for different layers. In detail for every graphical description technique in the ISO documents a transformation into a textual representation is given. Then a mapping from this textual representation to a set of sequences of service primitives is introduced. Each description part has as semantics a set of sequences of service primitives. As semantics of the combination the intersection of sets is chosen. As a consequence every description technique part describes one constraint or in an informal notion one aspect of a service property.

[2]In the following we do not explicitly mention the CCITT recommendations, if as a matter of the harmonization both ISO and CCITT documents are existing. Moreover, we use the phrase ISO documents as an abbreviation for ISO documents concerned with the service specification of a layer within the ISO/OSI basic reference model.

3 Results of the Formalization of the Specification Techniques

In this section we present some improvements of the description techniques used in ISO documents. One description technique we have examined are *time sequence diagrams (TSDs)* describing the allowed sequences of exchanged service primitives on both service access points. As a second description technique we focus on an isolated examination of sequences on one service access point, which is achieved in the ISO documents by automata and tables. We call these description technique *local constraints (LC)*. As a last description technique we have chosen the *queue model (QM)*, which describes the detailed exchange of service primitives on both service access points including parameters.

3.1 Time Sequence Diagrams

Time Sequence Diagrams (TSDs) are widespread graphical representations employed to clarify the communication between service users and a service provider in the OSI basic reference model. They are used in different documents to describe one property of a layers service. TSDs have been defined as a part of [ISO87a]. Later this ISO document has been harmonized by CCITT [CCI88b]. Both descriptions can be distinguished by some minor details. Nowadays a renewal of the TSD description has been done by ISO [ISO91a, ISO94]. In contrast to [ISO87a, CCI88b] one powerful operator and a special representation style has been moved into the appendix of the document, which is not a binding part. With the use of the results presented in this paper and [Fac95b, Fac95a] these parts can be transformed from comments to valid definitions. However, our main point of interest is [ISO87a, CCI88b] because they form the basis for the ISO/OSI basic reference model.

With TSDs timing precedences between service primitives can be expressed. In the following the term events is used instead of service primitives as an abbreviation.

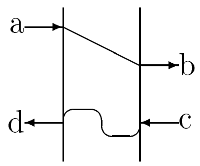

Figure 2: Example TSD

In Figure 2 a TSD describing the temporal ordering of the events a, b, c and d is presented. The two vertical lines which denote the Service Access Points (SAP) divide three different communication partners from the left to the right: User A (on the left-hand side), the service provider (between the two vertical lines) and user B (on the right-hand side). The vertical lines also denote an increase in time downwards. Therefore a temporal ordering of service primitives at each SAP is given. Two service primitives at different SAPs can be related by a solid line, which indicates a timing precedence. In Figure 2 the event b takes place after event a. Events connected with a tilde ($\sim$) are not related with respect to time. Therefore any temporal ordering between the events c and d in Figure 2 is allowed.

The TSDs of a service specification can be regarded as a description of the generation principle for the allowed sequences of events, which we call *scenarios*. To clarify the semantics of a TSD description of a service we introduce the notion *scenaric interpretation* which describes one possible scenario of a TSD without repetition. The *complete interpretation* describes an arbitrary number of repetitions of TSDs. The scenaric interpretation of a single TSD describes the allowed scenarios of events, but it is not possible that only a prefix of these scenarios take place. In other words the scenaric interpretation imposes the requirement that all events of a TSD have to take place.

3.1.1 Different Expressiveness of Two Styles

In [ISO87a, CCI88b] two different styles for describing TSDs are introduced. These Styles differ in the modeling of the service primitives' time consumption.

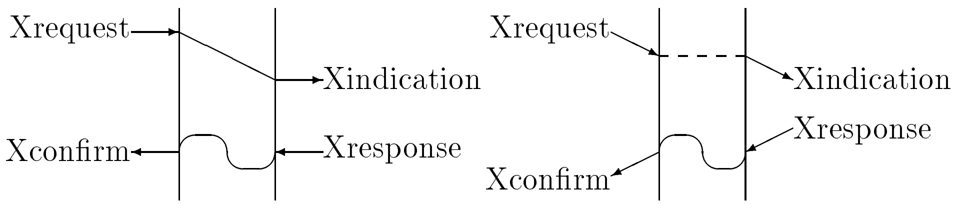

Figure 3: Different styles for TSDs

The left TSD presented in Figure 3 models the consumption of time in the service provider part. Therefore we name this style *service provider oriented*, in [CCI88b] this style is identified as style 1, in [ISO87a] as *logically correct presentation* or *preferred notation*. Because the time consumption is modeled in the service user parts we name the style used in the right part of Figure 3 as *service user oriented*. This style is named in [CCI88b] as style 2, in [ISO87a] as *alternative notation*. In [CCI88b] both styles for representing TSDs are regarded as equivalent:

> Each OSI Layer Service definition may adopt either of these styles for use in time-sequence-diagrams. No difference in the sequence of primitives being described is implied by the style used.

In [ISO87a] more careful words are used:

> Both presentations are intended to convey the same basic meaning.

However, both styles have a different expressiveness, which is not annotated in [ISO87a, CCI88b]. To show the different power of this two styles a TSD in a provider oriented style is introduced which cannot be transformed to the service user oriented style.

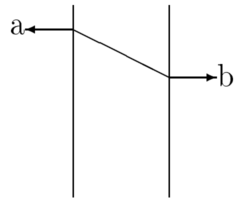

Figure 4: Only in the service provider oriented style presentable TSD

In Figure 4 a TSD in the service provider oriented style is shown. For this TSD is no equivalent representation in the service user oriented style existing. The reason therefore is the conflict between the time consumption and the orientation of the service primitives (input or output). This can be seen by the fact that in the service user oriented style the connection between the SAPs is only defined by a horizontal line. In Figure 4 therefore a slanted line is used. We refer to [Fac95b] for a more detailed discussion. As a result of the greater expressiveness the provider oriented style should be preferred. This fact delivers a formally based argument for the usage of the "logically correct presentation" in [ISO87a].

An interpretation problem of a TSD using the service user oriented style appears in the service specification of the physical layer [ISO88b, CCI88c].

Figure 5: TSD of the physical layer

The TSD presented on the left side in Figure 5 is used in [ISO88b, CCI88c]. The dotted line between the service primitives a and b suggests a timing precedence between them although it does not exists. This can be seen on the starting points of the arrows which are not in any timing relation. We suggest the use of the right side of Figure 5 to avoid a possible timing ambiguity between the service primitives a and b. Note that this possible misunderstanding is only based on the service user oriented style because it can not occur using the service provider oriented style.

3.1.2 Rigorous Interpretation

In nearly all ISO/OSI documents TSDs are only used to describe some examples of a possible service behavior. This can also be seen on the recent definition of TSDs [ISO94]:

> Time-sequence diagrams do not provide a complete and unambiguous description of an OSI-service. They are aimed at clarifying, mostly through the use of examples, the most complex aspects of a OSI-service.

Such a view is as a starting point for a formal oriented approach worthless, because all behaviors of a service provider are allowed. E.g. let a set of TSDs describe the behavior of a service provider. If a sequence of service primitives is not covered by this TSDs, the on examples based view allows that this sequence is also possible. As a consequence of this interpretation all sequences are possible. Therefore the used TSDs can be seen only as comments. To use formal methods we chose a rigorous approach: The used TSDs describe *uniquely* the possible behavior of a service provider. If a sequence of service primitives is not covered by at least one TSDs or a combination of TSDs, it is prohibited. Thus the set of TSDs describes only a set of allowed sequences, and further formal development steps can be carried out.

Another aspect of the interpretation is based on liveness properties. We claim that a TSD describes also a liveness property. All service primitives of a TSD have to happen and not only a prefix of the complete sequence. E.g. the scenaric interpretation of the TSD of Figure 4 only leads to the sequence $<a, b>$. This is necessary because during a data transfer the data has to be transfered or a disconnection has to happen. In other words the data request always leads to a reaction. This is expressed in our formal notation by the liveness part of the property.

3.1.3 Composition of TSDs

In the ISO documents TSDs are used as a collection of single TSDs. The TSD part of a service description can be seen as a set of single TSDs due to the graphical description, in which TSDs have been put side by side without explicit visible combination operation. Moreover, the combination of TSDs is not mentioned in the ISO documents. However, to define the semantics of TSDs an interpretation of the composition of TSDs is necessary. Therefore, we have chosen an or-operation which is associative and commutative like an union operation on sets. Although this operation is strong enough to express the composition of TSDs, the repetition of TSDs needs to be defined. For several reasons, e.g. the specification of a data transfer with reordering, we have chosen a repetition based on interleaving. The repetition of one TSD is achieved by an arbitrary number of interleavings of sequences which are the semantics of that TSD. For more details see [Fac95b, Fac95a].

3.1.4 Unrestricted Service User's Behavior

With TSD it is possible to restrict the behavior of the service users. This is a consequence of the fact that the service provider, which is the component being specified, is only responsible for the output service primitives. The input service primitives cannot be influenced by a service provider because they are produced by the environment, the service users. However, TSDs can urge input service primitives to happen.

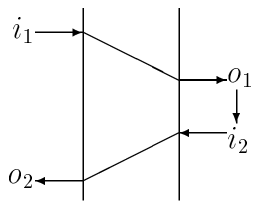

Figure 6: TSD with restriction of the service users behavior

The TSD presented in Figure 6 is taken from the specification of the *abracadabra service* [Tur93, ISO91b]. This TSD is also used within the service specification of all OSI layers with one difference: The arrow between the service primitives o_1 and i_2 on the right side of this TSD is only used in [Tur93, ISO91b]. We believe that this arrow is used to express a causality between the output service primitive o_1 and the input service primitive i_2. The reason for such a causality is that according to the TSD presented in Figure 6 after $<i_1, o_1>$ the input service primitives i_2 is expected. This is a restriction of the environment's behavior, that is responsible for the input event i_2. Such a restriction should be avoided [AL93]. To deal correctly with this situation the assumption/commitment style has been adopted for TSDs in [Fac95b, Fac95a]. Then the expectation of i_2 is described in the assumption part of the specification. Also a false input, e.g. if i_2 happens before o_1, can be dealt with. In both cases the environment has violated the rules first and therefore the component can behave arbitrary.

3.1.5 Absence of a Timing Precedence

TSDs use two elements for describing timing precedences. One of them is the tilde operator which describes that no timing precedence is existing.

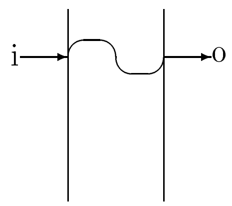

Figure 7: TSD with no timing precedences

The TSD presented in Figure 7 has been taken of the service definition of the data link layer [ISO88a, CCI88d]. In the definition of TSDs [ISO87a, CCI88b] the interpretation of this TSD is not provided. Due to the missing timing precedence between the events i and o the scenaric interpretation of the TSD of Figure 7 yields the sequences $<i, o>$ and $<o, i>$, if repetitions are ignored. Note that in that case a strange behavior can occur. The service provider produces the output element o and expects the input element i produced by the service user, who represents the environment of the component to be specified. Due to [AL93] the behavior should not be restricted. Therefore it has to be allowed that the service user does not react with an i, and the sequence $<o>$ has to be included in the set of sequences describing the scenaric interpretation of the TSD presented in Figure 7.

The unrestricted behavior of the service users makes an interpretation of the tilde operator used by TSDs possible. For more details see [Fac95b, Fac95a]. The interpretation problems based on the restricted service user's behavior appear in a later implementation. We believe that they are responsible for using the tilde operator only in the appendix of [ISO94]. In our semantics we give the tilde operator a clear and precise interpretation, which is not in contrast to later implementations.

3.1.6 Solved Interpretation Difficulty

A TSD used in the service specification of the network layer [ISO87b, CCI88e] and the data link layer [ISO88a, CCI88c] leads to an ambiguous interpretation.

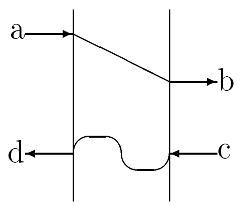

Figure 8: TSD of the network and data link layer

The TSD presented in Figure 8 allows two possible interpretations:

- Within the *restricted interpretation* both of the events a and b have to happen before an event d is possible.
- The *general interpretation* allows that event d happens before event b, because there exists no timing precedence between these two events in this TSD.

Note that all allowed sequences of the restricted interpretation are also included in the general interpretation. In the service specification of the network layer [ISO87b, CCI88e] the general interpretation is explicitly mentioned in the document. However, this additional text is missing in the service specification of the data link layer [ISO88a, CCI88d]. This leads to a possible ambiguity, whether the restricted interpretation can be chosen or not. We have chosen the general interpretation, because there is no timing precedence between the events a and d visible in the TSD presented by Figure 8.

3.1.7 Bidirectional Data Transfer

In no ISO document a bidirectional data transfer within one connection is possible, if the used TSDs are strictly interpreted. There is only one TSD given for the data transfer within one established connection. In this TSD only an unidirectional data transfer from the service user who has initiated the connection to his communication peer is possible. However, e.g. in the service specification of the transport layer [ISO84b, CCI88f] a simultaneous data transfer in both directions is allowed in the explaining text. To describe this behavior we suggest to mirror the TSD which describes the data transfer.

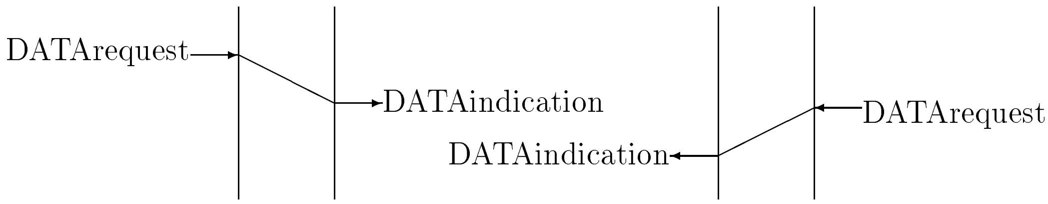

Figure 9: TSDs for bidirectional data transfer

In Figure 9 two TSDs are used to describe a bidirectional data transfer.

3.1.8 Incompleted TSDs

In nearly all service specifications of the OSI basic reference model a connection disruption which is a disconnect event issued by the service provider is possible. However, this fact is not specified by the used TSDs, only a textual addition is given in the ISO documents. In this textual addition it is only stated that every shown TSD can be disrupted by disconnect service primitives. The same observation can be achieved by the INRES service [Hog89, Hog91] which we use for simplicity.

Figure 10: Data transfer of the INRES service

Based on the TSDs presented in Figure 10 and with the exact interpretation, that every TSD has to be carried out completely, the following sequence of service primitives is possible. Suppose there have been two IDATreq, then a connection disruption occurs. Based on the TSDs of Figure 10 the service provider has to emit exactly two IDISind. However, most implementations provide in such a case only one IDISind.

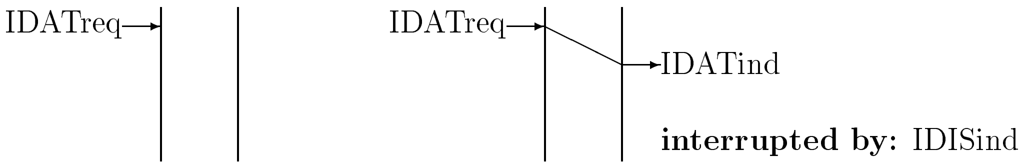

Figure 11: Modification of the INRES data transfer

The left TSD presented in Figure 11 is suggested in [BHS91] to deal correctly with a connection disruption. However, with this additional TSD a possible data lossage is also described, because then a IDATreq does not necessarily lead to a IDATind or IDISind. To avoid this we introduce a TSD, in which an interrupt can occur, presented on the right side of Figure 11. The semantics of this TSD is that every part of this TSD can be interrupted by an IDISind. For a formal semantics see [Fac95a, Fac95b]. So with this TSD a correct description of a connection disruption without data lossage is possible.

3.1.9 Modular Semantics for TSDs

In [Fac95b, Fac95a] we have given a modular semantics of TSDs. We have defined a mapping from an arbitrary single TSD to a set of all allowed sequences describing the correct behavior with respect to the TSD. We have also given a semantics of the composition of the single TSDs. With this technique it is possible to describe the formal semantics of the TSD part of a service specification in a modular way, because an interpretation of one TSD can be done without knowledge about the other TSDs. As an intermediate form of this transformation we used a specially defined simple process algebra. The new definition is necessary due to the nondeterminism operator normally used by process algebras. As an example let $<a, b>$ be the sequence described by one TSD and $<a, c>$

of another one. The combination of both TSDs expressed in a process algebra is:

$$(a.b) + (a.c)$$

However, in process algebras

$$(a.b) + (a.c) \neq a.(b + c)$$

holds. As a consequence the nondeterministic choice which TSD is described must be done before or at least at the same time as the event a happens. If the choice has been wrong, a deadlock appears even if the correct event would follow. E.g. the sequence $<a, b>$ is chosen after an a occurs and then follows the event c. However, to describe TSDs in a modular way the negation of the last formula should hold. To describe such a behavior in [BM94] the delayed choice operator has been introduced. In our used process algebra we introduced a nondeterministic operator with an equivalent behavior.

3.1.10 Service Primitives with Parameters

In all TSDs used for the service specification of the OSI basic reference model the parameters of service primitives are not included. E.g. the service primitive $DATreq$ is used instead of $DATreq(data)$. However, for modeling a communication the parameter $data$ is essential.

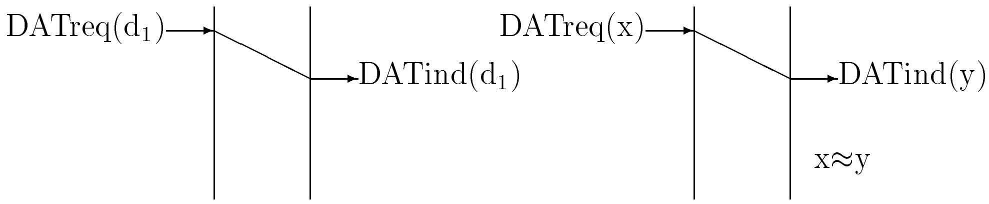

Figure 12: TSDs describing data transfer on behalf of parameters

The left TSD of Figure 12 uses service primitives with parameter. The interpretation of this TSD is that every $DATreq(d_1)$ leads to a $DATind(d_1)$. This is in contrast to the *residual failure probability* which is a part of the *quality of service* descriptions and is a measurement for non detectable failures. To describe this correct we introduce the concept of *parameter restriction*, presented in the right TSD of Figure 12. Here the equivalence relation $x \approx y$ is used to describe non detectable errors. With this concept even the negotiation of the quality of service parameters during the connection establishment phase is possible if the partial ordering relation $\leq$ is used instead of $\approx$.

3.1.11 General Points of Criticism

One problem of TSDs is that two service primitives taking place at two different service access points have to be related regarding time. For example due to the TSD shown in Figure 3 the event $Xrequest$ has to happen before a $Xindication$. But both service primitives are issued on different service access points. In a global setting it is possible that they happen on different locations in the world. The question that arises is how this two service primitives can be put in a timing relation. One solution is the use of a global time to which the local time can be transformed and the other way around.

Another point of criticism is that every communication event like a service primitive is in reality divided in a sending and receiving event. The exchange is done by handshake. The sender provides his information at the interface and expects a receiving event of the receiver. Later an abstraction of this exchange to one event can be done. However, using TSDs the splitting into two events is not possible like in other techniques, e.g. *Message Sequence Charts (MSC)* [CCI92].

3.2 Local Constraints

In the lower layers including the transport layer an isolated specification of the allowed sequences of service primitives at one service access point is done. We name this *local constraints (LC)* because only one service access point has to be examined without any possible interaction of the other one.

In the ISO documents two different description techniques are used for specifying local constraints. In all service specifications for describing local constraints finite state automata are used. This description technique we call *automata method*.

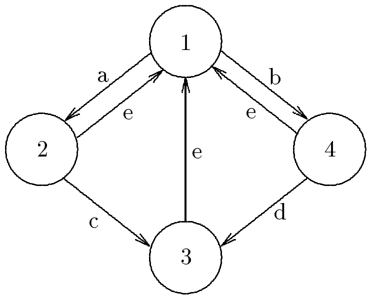

Figure 13: Automata method

The automaton presented in Figure 13 describes possible sequences of service primitives. E.g. after an a has happened only a c or e is possible. However, as a consequence of the used states it is possible that later implementations can be influenced [ISO84b, CCI88f]:

> The use of a state transition diagram to describe the allowable sequences of TS primitives does not impose any requirement or constraint on the internal organization of any implementation of the Transport Service.

To avoid this additionally to the automata method in the transport layer [ISO84b, CCI88f] a table based approach is used, which we call *table method*. Such a possible implementation is not restricted.

	This ↓				
can follow that ↓	a	b	c	d	e
a					+
b					+
c	+				
d		+			
e	+	+	+	+	

Table 1: Table method

Table 1 defines e.g. that after an a only c or e are possible. A closer examination shows that this table and the automaton presented in Figure 13 describe the same sequences. The major advantage of the table method is the higher abstractness which is a consequence of not using states.

3.2.1 Different Expressiveness

The table and the automata method are used in a similar way. Therefore the question of a equivalent expressiveness appears.

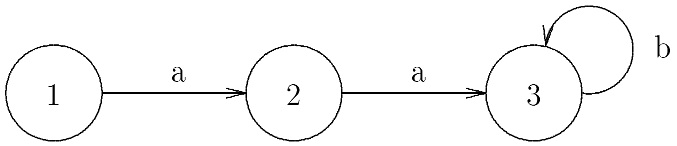

Figure 14: Not transferable automaton

The sequences of the automaton presented in Figure 14 cannot be described by a table. Using the table method there exist two possibilities for an entry for a as successor of a:

- a is allowed after an a has happened. Then also the sequence $<a, a, a>$ is accepted by the table method. But this sequence is not described by the automaton presented in Figure 14.

- a is not allowed after an a has happened. Then the sequence $<a, a>$ is not described by the table method. But this sequence is allowed by the automaton of Figure 14.

Thus the automata method has a higher expressiveness than the table method. This can also be demonstrated by the automaton used in the service specification of the physical layer [ISO88b, CCI88c] whose allowed sequences can not be described equivalently by a table. For details see [Fac95b].

As a consequence a dilemma arises: We should use the method with the highest abstractness, to avoid biasing later implementations. Therefore, the table method should be chosen. However, we have shown that this is not always possible. As a solution we suggest to use the table method whenever possible, and the automata method otherwise.

3.2.2 Unrestricted Service User's Behavior

With the description techniques used for LC a distinction between input and output service primitives is only derived by their naming. E.g. according to the table or automaton used in the service specification of the transport layer [ISO84b, CCI88f] the input service primitive T_DATArequest is not allowed at any time point. This service primitive is only possible during an open connection. As a consequence the service user is according to the used specifications restricted in its action. In [Fac95b] we extended the descriptions using the assumption/commitment style such that a free service user's behavior is possible.

3.3 The Queue Model

As a third description technique queues are used in the ISO documents. This technique we call *queue model*. The introduction of a third description technique is necessary because of the lacking expressiveness of the previous mentioned description techniques. The correct ordering of data during a data transfer can not be described by TSDs and LCs.

Example 1: Data Transmission Service Primitives

This example shows that a service provider preserving the sequential ordering of a data transmission can not be described using the description techniques for TSD and LC in combination.

Assume the possible scenarios if one user transmits successfully the data x_1 and later x_2:

$$<DATreq(x_1), DATind(x_1), DATreq(x_2), DATind(x_2)>$$
$$<DATreq(x_1), DATreq(x_2), DATind(x_1), DATind(x_2)>$$

If a specification would only use the description for TSD and LC, the sequence

$$<DATreq(x_1), DATreq(x_2), DATind(x_2), DATind(x_1)>$$

would also be described[3]. Then the sequential ordering of the data of the service primitives x_1 and x_2 is violated. □

Due to the above presented observations a third description technique is necessary. In the ISO documents therefore a technique is introduced which is based on FIFO structures. We call this technique *queue model (QM)*. As a first approximation the content of the used queues represents the non delivered data elements.

The queue model is not described in the ISO documents. It is the description technique with the least formal content because of the widely use of textual explanations.

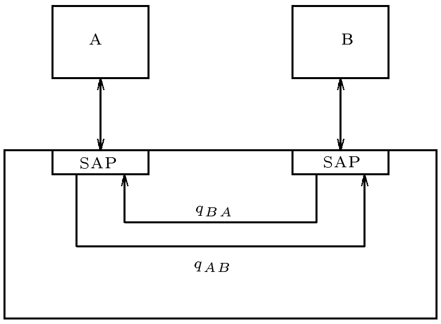

Figure 15: Queue model

In Figure 15 a queue model for a bidirectional connection is presented. In each queue objects are inserted or removed as a consequence of service primitives. E.g. the insertion of a data object $DO(x)$ is initiated by the service primitive $DATArequest(x)$.

	CO	DO	EXP	DIS
CO	N/A	–	–	DES
DO	N/A	–	A/A	DES
EXP	N/A	–	–	DES
DIS	N/A	–	–	DES

Table 2: Precedence table

As a second graphical description technique precedence tables are used by the queue model. We show in Table 2 an example of precedence tables. Here for simplicity CO denotes a connect object, DO a data object, EXP an expedited data object and DIS a disconnect object. The entry A/A (advance ahead) denotes that an expedited data object may overtake a normal data object in the queue. The intuition behind this is

[3] A formal proof of this property is given in [Fac95b].

that the expedited elements have a higher precedence than normal ones. The entry N/A (not available) describes e.g. that a connect object can not be followed immediately by another one. That a disconnect object may delete any preceding object is denoted by an DES entry.

3.3.1 Control of Activity

According to the ISO documents the main control is given to the service users with the exception of disconnect objects. This can be demonstrated by the service specification of the transport layer [ISO84b, CCI88f]:

> The objects which may be placed in a queue by a TS user (see §§12, 13 and 14) are:
>
> a) connect objects ...
>
> ...
>
> The only objects which can be placed in a queue by the TS provider are disconnect objects (representing $T - DISCONNECT$ primitives and their parameters).

This leads to the possible misunderstanding that the service users posses the activity control. In this case a service user may insert objects into the queue which are according to Figure 15 elements of the service provider. Due to the concept of data encapsulation such a behavior is not possible. Certainly we suppose that in the ISO documents it is only intended to give the service users the initiator role for insertion or deletion of objects. This event is under control of the service provider who is the only one with direct access to the queues.

3.3.2 Relation between Objects and Service Primitives

In the ISO documents no exact relation between service primitives and operations on the queue is given. This has been a major problem for deriving a formal semantics of the queue model. To provide a more formal notation than the used textual one we introduce the concept of *object tables*.

service primitive	object	operation	condition
CONNECTrequest$_A$	CO	add$_{AB}$	isempty(q_{AB})
DATAindication$_B$	DO	remove$_{AB}$	
Dummy		none	

Table 3: Object table

The first column of Table 3 contains the service primitive with an additional index for the denotation of the source of a service primitive. E.g. $_A$ is attached to the service primitives that appear at the service access point to the service user A. The second column is used for describing the object which is related to a service primitive. The third column contains the type of operation on the queues. We distinguish the operations:

add: An *add* operation indicates that the associated service primitive happens only if an object is appended to the queue.

remove: An *remove* operation indicates that the associated service primitive happens only if an object can be removed of the queue.

none: An *none* operation indicates that the associated service primitive happens without any effect to the queue.

The index is used for describing the queue in which the operation has to be done. E.g. $_{AB}$ stands for the queue from service user A to B. In the last column an additional condition for the operation can be given.

With this description an exact specification of the service provider's behavior is possible, because every service primitive precisely is related to an operation on the queues. This description is used as a basic for developing a formal semantics of the queue model [Fac95b].

3.3.3 Double Deletion

In the service specification of data link layer [ISO88a, CCI88d] an object for the synchronization is introduced. This object can be removed by a following reset object. This is indicated in the precedence table by a *DES* entry. In a textual addition it has been stated that then both objects, the synchronization and reset object are deleted. However, a *DES* entry describes only the deletion of the preceding object. To distinguish this different effects we suggest to use our *DESboth* entry, which indicates that both elements are removed.

4 Conclusion

In this paper we presented the results of a formalization of the ISO/OSI basic reference model, which are also valuable for informal descriptions. We demonstrated that the use of a constraint oriented specification style improves the reusability of a specification significantly. With this major improvement a formal specification of the different layers has been carried out in [Fac95b], leaving out only minor details.

The conclusions drawn from a formal specification are important for further development steps, e.g. a formal method for developing an implementation. But this formalization is also valuable for the informal descriptions in the actual and further documents which describe a service specification, because the used techniques gain more clarity. As an example we have significantly improved time sequence diagrams. We have introduced an exact interpretation of TSDs such that some elements can now be used for further formal work. We have given the tilde symbol an exact interpretation. With our presented method, the tilde symbol can be moved from the appendix of [ISO94] to the main part of this definition. Furthermore the text of the definition of TSDs can be significantly enlarged, because of the existing precise interpretation of TSDs. This would also lead to more clarity by exchanging TSDs between several users. In the definition of TSDs [CCI88b] the two used styles have been suggested to be equivalent but we have shown that these styles have a different expressiveness. As a consequence we provided a formal argumentation for using the in [ISO87a] called "correct representation" because of its higher expressiveness. We defined the semantics of composing TSD, which has not been introduced by the ISO and CCITT texts. We have also detected and corrected some lack of preciseness in the graphical representation of TSDs.

Despite the fact that the description techniques used for the specification of the local constraints have a very high formal level we have shown the different expressiveness of the used description techniques. This leads to a contrary situation: The more abstract technique can not be used in every layer.

The third description technique used for the queue model gains more preciseness even in the informal notation by the introduction of the object tables. Furthermore we corrected the use of the precedence table in the case of double deletion.

Acknowledgment

I thank Manfred Broy, Ekkart Rudolph and Ketil Stølen for careful reading preliminary versions of this paper and providing valuable feedback.

References

[AL93] Martin Abadi and Leslie Lamport. Conjoining specifications. Technical Report 118, Digital System Research Center, December 1993.

[BHS91] Ferenc Belina, Dieter Hogrefe, and Amardeo Sarma. *SDL with Applications from Protocol Specification.* Prentice Hall, 1991.

[BM94] J. C. M. Baeten and S. Mauw. Delayed choice: an operator for joining Message Sequence Charts. In Dieter Hogrefe and Stefan Leue, editors, *Participant's Proceedings of the Seventh International Conference on Formal Description Techniques (FORTE 94)*, pages 327–341, 1994.

[CCI88a] CCITT. X.200, reference model of Open System Interconnection for CCITT applications. Blue Book, FASCICLE VIII.4, Recommendations X.200-X.219, November 1988.

[CCI88b] CCITT. X.210, Open System Interconnection layer service definition conventions. Blue Book, FASCICLE VIII.4, Recommendations X.200-X.219, November 1988.

[CCI88c] CCITT. X.211, physical service definition of open system interconnection for CCITT applications. Blue Book, FASCICLE VIII.4, Recommendations X.200-X.219, November 1988.

[CCI88d] CCITT. X.212, data link service definition of open system interconnection for CCITT applications. Blue Book, FASCICLE VIII.4, Recommendations X.200-X.219, November 1988.

[CCI88e] CCITT. X.213, network service definition of open system interconnection for CCITT applications. Blue Book, FASCICLE VIII.4, Recommendations X.200-X.219, November 1988.

[CCI88f] CCITT. X.214, transport service definition for Open System Interconnection for CCITT applications. Blue Book, FASCICLE VIII.4, Recommendations X.200-X.219, November 1988.

[CCI92] CCITT. Message sequence chart (MSC). Recommendation Z.120, November 1992.

[Fac95a] Christian Facchi. Formal semantics of time sequence diagrams. Technical Report TUM-I9540, Technische Universität München, 1995.

[Fac95b] Christian Facchi. *Methodik zur formalen Spezifikation des ISO/OSI Schichtenmodells.* PhD Thesis, Technische Universität München, 1995. Published in Herbert Utz Verlag Wissenschaft, München (ISBN 3–931327–94-9).

[Hog89] Dieter Hogrefe. *Estelle, LOTOS und SDL.* Springer-Verlag, 1989.

[Hog91] Dieter Hogrefe. OSI formal specification case study: the Inres protocol and service. Technical Report IAM-91-012, Universität Bern, 1991.

[ISO84a] ISO. ISO 7498: Information processing systems - open systems interconnection - basic reference model, 1984.

[ISO84b] ISO. ISO DP 8072: Information processing systems - open systems interconnection - transport service definition, 1984.

[ISO87a] ISO. Information processing systems - Open Systems Interconnection - service conventions. Technical Report ISO TR 8509, ISO, 1987.

[ISO87b] ISO. ISO 8348: Information processing systems - open systems interconnection - network service definition, 1987.

[ISO88a] ISO. ISO 8886: Information processing systems - data communication - data link service definition, 1988.

[ISO88b] ISO. ISO DP 10022: Information processing systems - open systems interconnection - physical layer service definition, 1988.

[ISO89a] ISO. LOTOS, a formal description technique based on the temporal ordering of observational behaviour. ISO International Standard 8807, February 1989.

[ISO89b] ISO/IEC. Information technology - Open System Interconection - LOTOS description of the session service. Technical Report ISO/IEC/TR 9571, International Organization for Standardization Geneva, 1989.

[ISO91a] ISO. Revised text of CD 10731, information technology - Open Systems Interconnection - conventions for the definition of OSI services. Technical Report ISO/IEC JTC 1/SC 21 N 6341, ISO, 1991.

[ISO91b] ISO/IEC. Information technology - Open System Interconection - guidelines for the application of Estelle, LOTOS and SDL. Technical Report ISO/IEC/TR 10167, International Organization for Standardization Geneva, 1991.

[ISO92] ISO/IEC. Information technology - telecommunications and information exchange between systems - formal description of ISO 8072 in LOTOS. Technical Report ISO/IEC/TR 10023, International Organization for Standardization Geneva, 1992.

[ISO94] ISO. Final DIS text of ISO/IEC 10731, information technology - Open Systems Interconnection - conventions for the definition of OSI services. Technical Report ISO/IEC JTC 1/SC 21 N 8604, ISO, 1994.

[Tur93] Kenneth J. Turner, editor. *Using Formal Description Techniques.* John Wiley & Sons, 1993.

[Vis88] C.A. Vissers. FDTs for distributed systems. In R. Speth, editor, *Research Into Networks and Distributed Applications*, pages 39–49. North-Holland, 1988.

[VL86] Chris A. Vissers and Luigi Logrippo. The importance of the service concept in the design of data communication protocols. In *Protocol Specification, Testing, and Verification*, volume V, pages 3–17, 1986.

[VSvSB91] Chriss A. Vissers, Giuseppe Scollo, Marten van Sinderen, and Ed Brinksma. Specification styles in distributed systems design and verification. *Theoretical Computer Science*, 89:179–206, 1991.