

- -
AP IX-43-E
ANNEX A

(to Recommendation X.290, Part 1)

Options

- A.1 Options are those items in a Recommendation* for which the implementor may make a choice regarding the item to suit the implementation.
- A.2 Such a choice is not truly free. There are requirements which specify the conditions under which the option applies and the limitations of the choice.
- Conversely, there may be mandatory or conditional requirements, or prohibitions, in a Recommendation* which are dependent on the choice made or on a combination of the choices already made.
- A.3 The following are examples of options and associated requirements; the list is not exhaustive:
- a) "Boolean" options: the option is "do or do not do"; the requirement is "if do, then do as specified."
 - b) Mutually exclusive options: the requirement is to do just 1 of n actions, the option is which of them to do.
 - c) Selectable options: the option is to do any m out of n actions, with a requirement to do at least one action ($1 \leq m \leq n$ and $n \geq 2$).
- A.4 Options may apply to anything within the scope of a Recommendation* (e.g. static or dynamic aspects, use or provision of a service, actions to be taken, presence/absence or form of parameters, etc.).
- A.5 Another type of distinction is between service-user options and service-provider options.
- A.6 In a wider context, the choice will be determined by conditions which lie outside the scope of the Recommendation* (e.g. other Recommendations* which apply to the implementation, the protocols used in the (N+1) and (N-1) layers, the intended application, conditions of procurement, target price for the implementation, etc.). However, these have no bearing on conformance to the Recommendation* in which the option appears.

ANNEX B
(to Recommendation X.290, Part 2)

Guidance for protocol Recommendations* writers
to facilitate conformance testing

B.1 Introduction

This annex gives guidance, primarily for the specifiers of new protocol Recommendations*, to facilitate conformance testing by ensuring a very clear understanding of the conformance requirements.

B.2 Guidance on scope and field of application

B.2.1 Precision in the scope and field of application sections sets the tone for precision in the rest of the Recommendation*. The requirements stated in the Recommendation* should be consistent with the scope and field of application and vice versa.

B.2.2 The scope should distinguish clearly between the following three types of information included in the protocol Recommendation*:

- a) the definition of the procedures for communication to be followed at the time of communication;
- b) requirements to be met by suppliers of implementations of the procedures;
- c) guidance on how to implement the procedures.

Guidance on how to implement the procedures does not constitute additional requirements nor does it have any bearing on conformance. If such guidance is included, the scope should make these points clear and indicate how guidance can be distinguished from the requirements of the Recommendation*. This distinction is much easier to make if guidance is separated from requirements. The recommended method of such separation in the ISO Directives is to place guidance in notes and annexes.

B.2.3 It should be clear to whom the Recommendation* applies.

B.2.4 It should be clear at what time the Recommendation* applies.

Protocol procedures apply between pairs of communicating parties at the time of communication. If there might be any ambiguity over which communicating parties are involved, this should be resolved in the scope.

It is best if protocol Recommendations* are written in such a way that the requirements are to be met by a single communicating party (the "first" communicating party for this purpose) for the benefit of one or more other communicating parties (the "second" communicating parties). Then when two (or more) communicating parties are all expected to communicate in conformance with the Recommendation*, the Recommendation* is first applied to one party, treating it as the "first" one, and then to the other(s) in turn. This ensures that if the procedures are violated, it is clear which party is at fault.

B.2.5 If any guidance is given about factors which are not standardized definitively, the scope should make it clear that any such guidance can be ignored without affecting conformance.

B.2.6 The aspects which are excluded from the scope should be identified clearly.

Not all factors relevant to the procedures or to products which implement them need to be standardized; indeed it is often desirable to leave some implementor freedom. For instance, it may be desirable to omit in a protocol Recommendation* any requirements for explicit values of timeouts, but to give guidance instead.

The scope should make it clear which aspects are standardized definitively, which are covered by guidance but not by any requirements, and which are excluded from consideration by the Recommendation*. Any aspects which one might think would be covered, on the basis that they are closely related to aspects which are standardized, need explicit mention.

B.2.7 All options should, if possible, be identified clearly in the scope.

Options are one of the most troublesome, but unfortunately necessary parts of protocol Recommendations*. They fall somewhere between what is standardized and what is not. They will be covered in greater depth below. At this point, what is important is that options are not buried deep inside the Recommendation* but are declared clearly at the beginning. If the number and detailed nature of the options makes this impractical, one should seriously ask whether such complexity is really necessary. Can detailed options be grouped together in some way (e.g. classes) to simplify the Recommendation*?

B.2.8 The scope and field of application should be reviewed after considering the rest of the Recommendation*.

It is often not possible to satisfy some of the above suggestions until the rest of the Recommendation* has been considered. Therefore, it is generally necessary to return to the scope, to check that it really does agree with the contents of the Recommendation*. It is common to find that sections quite outside the scope have found their way in.

B.3 Guidance on references

B.3.1 OSI protocol Recommendations* should refer to the OSI reference model, the relevant service Recommendations* and to any relevant Recommendations* for protocol conventions, guidelines, or formal description techniques.

B.3.2 It should be made clear whether conformance to the protocol Recommendation* requires conformance to any part of any other Recommendation*.

B.3.3 It should be made clear whether each reference is to a particular version of the referenced Recommendation* or to each successive version.

Normally, the latest version is required, but this can cause problems as changes to the other Recommendation* might affect conformance to this Recommendation*.

B.4 Guidance on requirements and options

B.4.1 The status of every requirement should be unambiguous.

Since optional and conditional requirements are so common, there is a tendency to interpret everything which can be interpreted as optional as being optional.

B.4.2 It should be possible for an instance of communication to conform with all the mandatory dynamic conformance requirements.

B.4.3 The conditions under which conditional requirements apply should be spelt out clearly.

B.4.4 It should not be impossible for the implementor or supplier to know what these conditions are.

B.4.5 There should be no possibility of confusion between what is optional dynamically and what is optional statically.

There may be mandatory static conformance requirements for the support of features whose use at the time

of communication is optional. Conversely, a message whose use is mandatory in a given context at the time of communication may be part of a protocol mechanism whose support is optional statically.

B.4.6 If the Recommendation* contains a "shopping list" of options, and there are restrictions on the allowed combinations of such options, then the restrictions should be specified clearly. These should include identification of any mutual exclusions and any minimum and maximum limits to the allowed range of options.

B.4.7 If the Recommendation* does not give any rules for selection of options, it should be made clear in the scope that only the total range and individual options are standardized, but not the selection.

B.4.8 Legitimizing options should be avoided. These are options which allow alternative and incompatible versions of the same thing to claim conformance to the same Recommendation*. While they do not of themselves prevent an objective understanding of conformance, they may frustrate the aims of OSI.

B.4.9 There should be no options which give the implementor permission to ignore important requirements of the Recommendation*. Such options devalue the Recommendation* and the meaning of conformance to it.

B.4.10 If there are prohibitions in the Recommendation*, they should be sufficiently precise to be meaningful.

Many Recommendations* have sections which say in effect "do all of this and nothing else". Such prohibitions may be meaningless, because every protocol conveys some information which is not standardized, the so-called "user data", and every standardized product has attributes which are not standardized, e.g. weight. It may be difficult to draw a clear objective line between things the Recommendation* cannot forbid and those which the writers of the Recommendation* want to forbid, unless the prohibitions are stated explicitly.

B.5 Guidance on Protocol Data Units

B.5.1 The permitted set of PDU types and parameter encodings should be stated clearly.

B.5.2 The permitted range of values should be stated clearly for each parameter.

B.5.3 All values, which fall outside the stated permitted range, should be stated explicitly to be invalid.

If not, some people will argue that such values are undefined but allowed, whilst others will argue that they are invalid.

B.5.4 It should be clear whether or not undefined PDU types are allowed.

It is safer if all undefined PDU types are declared to be invalid.

B.5.5 Critical undefined values should be mentioned explicitly in the scope as being undefined.

B.5.6 There should be a defined procedure to be followed by the first communicating party in each case of it receiving an invalid or undefined PDU type or parameter.

B.5.7 It should be possible to detect whether the defined procedure has been followed in such cases. If it is not, then it should be because it does not matter.

Sometimes the procedure to be followed upon receipt of an invalid PDU is intentionally the same as when

some valid PDUs are received in the same circumstances. For example, the procedure might be to do nothing until one specific type of PDU is received, everything else being ignored. In such cases, it probably does not matter that the error has apparently gone undetected. In some other cases, the intention may be that error cases should be given special treatment, but that the procedure has been poorly chosen with the result that it cannot be distinguished from the action in non-error cases.

B.5.8 If, in the encoding of PDUs, there are any fields declared to be reserved, then there should be a clear statement of what values, if any, are allowed or disallowed in these fields.

B.5.9 If interrelated parameters can be carried on separate PDUs, then the set of permitted relationships between the values of these parameters should be precisely and clearly defined.

B.5.10 If the parameter encoding allows for parameters to be specified in any order, and the PDU format places restrictions on the permitted orders, then these restrictions should be clearly stated. It should be recognized that if many different orders are permitted, then a large representative sample of different orders ought to be tested. The added complexity of testing conformance should, therefore, be adequately compensated by some advantage in allowing this freedom.

B.5.11 The order in which the bits, octets, etc. should be carried in the underlying protocol should be stated clearly.

For example, should a two octet integer travel most or least significant octet first? It is surprising how often such simple causes of ambiguity are overlooked.

B.5.12 The relationship between SDUs and PDUs should be defined clearly.

B.6 Guidance on states

B.6.1 Protocol procedures are often defined using a finite state approach, whether formalized or not. The specification of these states is often incomplete.

B.6.2 Each state should be defined clearly.

B.6.3 If there are events which can only occur in a subset of the possible states, then possible occurrence of an event should be distinguished from valid occurrence.

B.6.4 The required actions and state transitions should be defined for each possible state/event pair. In particular, they should be defined for possible but invalid state/event pairs.

B.7 Guidance on Formal Description Techniques

B.7.1 The following requirements apply only to those Recommendations* which include a formal description. Precise, unambiguous Recommendations* can be written without the aid of a formal description technique (FDT), but in complex Recommendations* such as protocols formal descriptions are highly recommended. It should, however, be realized that they can create problems themselves in relation to conformance.

B.7.2 It should be clear whether the formal description forms an essential part of the Recommendation* or is provided only for guidance.

It is very important to have a clear understanding of the status of the formal description. Ideally there should be no discrepancies between the text and the formal description, but because this is very difficult to achieve in practice it is important that the reader knows which takes precedence. If the formal description is provided only for guidance, it cannot

define conformance requirements.

B.7.3 The FDT should be well-defined, stable and properly referenced.

B.7.4 If the formal description defines requirements, but not all the requirements of the Recommendation*, then it should be stated clearly that the text includes requirements which are not covered by the formal description and these additional requirements should be identified clearly.

B.7.5 If the formal description defines requirements, and it also defines an allowed way of implementing some aspects of the protocol, but there is intended to be freedom for the implementor to implement those aspects in some other way, then this constitutes over-definition. This is all too common in formal descriptions, and creates difficulties in relation to conformance. If the formal description is an essential part of the Recommendation*, then text should be provided to qualify it, indicating where such over-definition exists and what the real requirements are.

The problem usually arises because the formal description describes the internal behaviour of an idealized implementation, rather than the observable external behaviour that is required. It is only the observable external behaviour which can be tested, and therefore it is only this which should constitute requirements for conformance purposes. It may well be that a different FDT should be used for defining the requirements from that used to provide guidance to implementors.

B.8 Miscellaneous guidance

B.8.1 Information which may appear obvious should nevertheless be stated.

If something is omitted because it is "obvious", some readers will assume it is required because it is "obvious", but others will assume that it is omitted to provide freedom for implementors. For example, does the existence of a checksum imply that it has to be checked?

ANNEX C

(to Recommendation X.290, Part 2) Incomplete static conformance requirements

C.1 As a matter of historical record, the development of protocol Recommendations* has been undertaken in parallel with the determination of the meaning of conformance, and in particular with the gaining of understanding of the distinction between static and dynamic conformance.

C.2 Therefore, some early protocol Recommendations* will not give a complete specification of the requirements for static conformance. Typical of the incompleteness is the requirement to support a particular function without saying whether this applies to sending, receiving or both; or the absence of precise conditions of detection of protocol errors in received incoming messages.

C.3 Accordingly, there may be different interpretations of what a conforming implementation is.

C.4 In future Recommendations* or at the time of revision of existing Recommendations* it will be necessary to provide a thorough specification of static conformance requirements. This would include the specification of conditions applicable to the implementation or non-implementation of everything that is neither always mandatory nor always optional.

C.5 In the short term, it is essential that, at very least, current drafts are modified to clarify the present situation: it is not considered acceptable that Recommendations* should be issued in a form in which implementation requirements suffer from extensive ambiguity. Consideration also needs to be given to what should be done where the protocols have already reached the status of ISO International Standard or CCITT Recommendation.

C.6 No other short-term solution is seen other than to accept and state clearly that all capabilities not covered explicitly in the static conformance requirements are optional; and to minimize the potential problems that this may cause by specifying that:

- a) only implementations which
 - 1) implement everything that is explicitly specified as mandatory; and
 - 2) do not omit anything unless it is explicitly stated to be optional, even though there may be a
- (3184)

- -
AP IX-43-E

general clause of the sort "if not specified, then optional";

are to be designated as "conforming" without qualification;

b) any implementation which

1) implements everything that is explicitly specified as mandatory; and

2) omits things which are not explicitly stated to be optional, perhaps because of a general clause of the sort "if not specified then optional";

shall be described as conforming to a subset.

C.7 Implementations which omit anything that is mandatory do not conform at all.

Note - A system conforming without qualification will not necessarily interwork with another system, nor will it necessarily work better than a system conforming to a subset. The "perfect" system may refuse from other systems PDUs which it would consider as incorrect or incomplete. Thus, it may reject or abort connections.

Therefore, special consideration should be given to conformance with respect to the detection of protocol errors, especially when this detection can be explicitly or implicitly optional.

- -
AP IX-43-E

- -
AP IX-43-E