

This document represents the extent of agreements reached during a combined meeting in Cambridge between SC22/WG11 and SC21/WG8/RPC on the issues raised in the liaison statement from WG11 to SC21/WG8/RPC (WG11/N353).

Note that because RPC adopted some one of several comments on notation for Real types and values, there is now a difference in that syntax, which is not covered here.

Item 2.1: Form of the Grammar

Resolution: Form of the grammar is an editorial matter for the two standards. What is important is that the RPC grammar be shown to be a subset of the "global" IDN found in LIPC. To this end, an informative annex may be added to DIS 11578-2, if there is time. In addition, WG11 will make an effort to remove extraneous productions from LID/LIPC in order to facilitate the derivation.

Item 2.2: Reserved words.

Resolution: Agree that reserving some keywords is necessary to facilitate parsing. Only those keywords which, for syntactic reasons, cannot be handled as predefined identifiers (primarily type names) should be reserved. A complete list will be developed when the grammar(s) are finalized.

Item 2.3: Annotations/Extensions

Resolutions:

- a. Agree to use square brackets ([]) as delimiters. Both standards will incorporate text (an annex?) defining trigraph substitutes for square and curly brackets ({ }) for national character sets which do not contain these characters. The trigraphs introduced in the C standard (ISO 9989) will be used.
- b. RPC will consider using this text.
- c. RPC will consider changing the term "extension" to "annotation", although this only affects the exposition and not the IDN language.
- d. RPC will reference ISO 8824 for definition of object-identifier value notation; LIPC/LID already does (although it repeats the definition). The braces will not be omitted.

Item 2.4: Interface-identifiers.

Resolution: documents prior agreement.

Item 2.5: Type declarations.

Resolution: documents prior agreement.

Item 2.6: Character types.

Resolutions:

- a. RPC agrees in principle. ISO 2375, however, may represent an exception.
- b. RPC will allow any of: object-identifier or ISO_10646 collection-name or ISO_2375 registry-index.
- c. A better solution might be to make
collection-identifier = "ISO_10646" collection-name |
"ISO_2375" registry-index . This problem needs further study, but we agree in principle.
- d. The proposed text will be considered. It is accepted in principle, but allowance for multiple (8-bit) character sets, switched by escape-sequence (ISO 8824 GeneralString) must be supported.

Item 2.7: Character type syntax error.

Resolution: accepted by RPC.

Item 2.8: Interface-reference.

Resolution: documents prior agreement.

Item 2.9: Choice-type.

Resolutions:

- a. The change of the type name is restricted to the RPC text. It does not affect alignment.
- b. Agreed that the LIPC syntax will be:
choice-type = "choice" "(" [field-identifier ":"] tag-type
["=" discriminant] ")" "of" "(" alternative-list ")" .
discriminant = value-reference . RPC will disallow the field-identifier and require the discriminant, making it a proper subset.
- c. LIPC will revise alternative to:
alternative = "(" select-list ")" [field-identifier ":"]
type-specifier . RPC will require the field-identifier, making it a proper subset.
- d. RPC will consider requiring "default" to be last, although the committee does not consider there to be a good reason for this restriction. LIPC will continue to require "default" to be last, in order to honor its agreement with SC22/WG17.

Item 2.10: Require alternatives to cover the tag-type.

Resolution: Abandoned. Both standards will state that the actual tag-type will be the subtype comprising the tag-values actually (or implicitly) occurring in the alternative-list, unless a "default" is specified, and that the "runtime" occurrence of any other value shall be a marshalling error.

Item 2.11: Termination Parameters.

Resolution: Agree in principle, but the offered text is poor. The scope of parameter names is the list in which they occur, and in this regard the argument-list for a given procedure(-type) is distinct from any and all termination-lists which may also apply to that procedure(-type). The same parameter name may appear in two such lists but no significance is attached thereto - they are distinct and unrelated parameters.

Item 2.12: Unaliased pointers

Resolution: Agree in principle. In RPC "unaliased" deals with the relationships between two pointer values which are (components of) parameters on the call and/or on the return. LIPC will support the keyword in pointer-types, but ascribe a larger meaning, as indicated in the WG11 comment. LID will not discuss the concept. None of the standards will support the notion that it is possible to construct a pointer-value which refers to a component (only) of an aggregate-type.

Item 2.13: Restricted pointers

Resolution: LIPC will permit the keyword "restricted" in pointer-types, but state that its meaning is "unaliased, excluding(null)". WG11 believes that use of the keyword should be deprecated.

Item 2.14: Value-expressions

Resolutions:

- a. LIPC will be changed to have
value-identifier = [interface-identifier "::"] identifier . LIPC will otherwise incorporate LID value-expression productions. An effort will be made to use the term value-reference in LIPC if this can be reconciled with the needs of LID.
- b. RPC will consider modifying the production for value-reference to indicate that the interface-identifier is only allowed in a reference to a value-identifier, and the ". identifier" construct is only allowed in what LID calls a dependent-value, and these are mutually exclusive. It is recognized that, since all of the LID objects value-identifier, parametric-value and dependent-value may produce simply "identifier" the LID productions produce shift/reduce conflicts in Yacc parsers, which RPC wishes to avoid.

Item 2.15: Object-identifier

Resolution: Agreed that object-identifier-type and object-identifier- value should be first-class types and values in RPC as well as LIPC. The syntactic suggestions will be considered.

NEW Item 2.16: Names qualified by interface-identifier.

Resolution: LIPC productions for value-reference (or value-identifier) and type-reference (or defined-type) will permit interface-identifier "::". This notion is inappropriate to LID.

NEW Item 2.17: IN/OUT by field.

Resolution: Both working groups agree that this idea has merit, but the impact is too complex and not well enough understood to incorporate in the initial versions of these standards.

NEW Item 2.18: Scope of enumeration/state values.

Resolutions:

- a. Both standards will define the scope of enumeration value identifiers (literals) to be the interface-type. This ensures unambiguous referenceability, although it may constrain some uses. This rule may be relaxed in future addenda to the standards.
- b. LID/LIPC will be changed to remove the notion "qualified-value", which was needed only to disambiguate state/enumeration literals and creates syntactic ambiguities.
- c. Further work on scope of names is needed and will be undertaken jointly for the DISs.

NEW Item 2.19: Subtype syntax

Resolution: The colon separating "base" types from "subtype-specifiers" in all three standards will be deleted. The subtype-specifier keywords are reserved, making the colon unnecessary. Moreover, use of the colon to separate optional field- and argument-names from their datatypes as well as the subtype separator causes reduce/reduce conflicts (LR(1) problems) in IDN parsers.

NEW Item 2.20: Default character set.

Resolution:

a. LIPC will accept the RPC syntax change to permit a default to be specified in the interface-type "header".

b. LIPC will say that in the absence of a repertoire-identifier and the absence of a default repertoire declaration the default repertoire is implementation-defined. RPC will say the same, possibly adding a note that the default may be somehow defined to the process which maps the IDN to the association contract. There is no specified mechanism for this.