Subject: Resolution of comments on ballot on CD 13886 - Language Independent Procedure Calling
Source: SC22/WG11
Date: April 1994
References: SC22/WG11 N376R - CD 13886 (SC22 N1507)
SC22/WG11 N394 - Summary of voting and comments (SC22 N1572)

UK-1 Add "global data", do not add "global symbol".
Text to be supplied

UK-2 Accepted in principle, UK to provide text

UK-3 Accepted; move 3.2 to 3.31, and renumber 3.3 through 3.31

UK-4 Accepted; replace "<image, association>" by "<procedure image, association>".

UK-5 Accepted.
Rewrite as follows:

4.1 Modes of Conformance.

An information processing entity claiming conformance to this International Standard shall conform in either or both of the following ways.

4.1.1 Client mode conformance

In order to conform in client mode, a language processor shall allow programs written in its language to call procedures written in another language and supported by another processor, using the language-independent procedure calling (LIPC) as provided by clauses 5, 6 and 7 of this International Standard. In this case it is said to conform in (and be able of operating in) client mode. As part of this, the language processor shall define a mapping from its own procedure calling model to the LIPC model.

<<followed by the Note from 4.1.1>>

4.1.2 Server mode conformance

In order to conform in server mode, a language processor shall allow programs written in another language to call procedures written in its language (i.e. it will accept and execute procedure calls generated by another processor which is executing in a program that is written in that other language and which is operating in client mode, and return control to that client processor upon completion), using the language-independent procedure calling (LIPC) as provided by clauses 5, 6 and 7 of this International Standard. In this case it is said to conform in (and be able of operating in) server mode. As part of this, the language processor shall define a mapping from the LIPC model to its own procedure calling model.

<<Insert here the Note from 4.1, followed by the original Note from 4.1.2>>

UK-6 Accepted; leave to editor.

UK-7 Accepted. Title of clause 5 becomes "An informal model of procedure calling". Title of clause 6 becomes "A formal model of procedure calling".

UK-8 Accepted

UK-9 Accepted in principle: replace "International .. Datatypes" by "ISO/IEC 11404 Languages-Independent Datatypes"

UK-10 Accepted. Also, replace last word of the sentence ("exception") by "error".

UK-11 Not accepted, comment too vague.

UK-12 Accepted

UK-13 Accepted (is in 1st note)

UK-14 Accepted (is in 5.2.1.3, last note, last sentence)

UK-15 Accepted, reword sentence. Text from Jon

UK-16 In 3rd sentence: remove "for LIPC", replace "shall" by "may". Resolution: remove references to partitioning? Implementations conforming to this International Standard may support a mechanism for the sharing of global data and may support partitioning of global data. If the implementation supports either of these mechanisms, they shall be implementation-defined. Leave open.

UK-17 Accepted

UK-18 Accepted. The word restricted should be in the syntax type font.

UK-19 Accepted.

UK-20 Accepted. Order should be: normal, ab-normal, cancel.

UK-21 Accepted: External Cancellation. First sentence as proposed (with "external cancellation").

In 5.3.1, replace the bullet list with:

- normal termination of a procedure invocation returning the output parameters, input/output parameters, and result (if any)
- abnormal termination, in which the procedure itself detects an error or other unusual condition
- external cancellation, in which some other entity determines that the procedure should terminate
- hardware or software detected events which may or may not be critical to the proper execution of the application
- asynchronous events or notification
- type or value mismatches in parameter passing or return
- failure of the underlying invocation service itself.

UK-22 Accepted.

UK-23 Accepted.

UK-24 Accepted.

UK-25 Accepted. The change of title (UK-7) clarifies the purpose of the section.

UK-26 Accepted in principle. Alternatives are invited.

UK-27 Accepted, first sentence will read: "A box is a generic term for a container that holds a value of a particular datatype, for example what, in some contexts, would be called a "variable"."

UK-28 Accepted

UK-29 Accepted. Add immediately after the line "read: Box -> Value":

The above three lines are called signatures <<italicise>>. Each signature lists the name of an operation, the types of the inputs (if any) of that operation, and the types of its outputs (if any). An x <<special font>> (the cartesian product operator) separates input (or output) types.

UK-30 Accepted, use "unspecified".

UK-31 Accepted in principle. Alternatives are invited.

UK-32 Accepted. Top of page 14. Text "x" should be a "cross symbol"

UK-33 Answer is yes. In 6.3, replace the first two bullet items by the following two bullet items:

- Global symbols are used to refer to values that are permanently associated with the procedure (e.g., other procedures, non-local variables, or ‘‘own’’ variables).
- Local symbols are used to refer to values that exist only for the duration of a single invocation (e.g., the local ‘‘stack frame’’ variables).

Add a new last note to section 6.3:

Note - By binding global symbols to boxes, these global symbols can (indirectly) refer to values created at arbitrary times, and be associated with the given procedure for arbitrary periods. Thus the phrase ‘‘permanently associated’’ above is not a substantive restriction to what can be modelled.

UK-34 Is in 6.3, 2nd bullet. Reworded by UK-33

UK-35 Accepted

UK-36 Accepted. Reword 1st sentence to "A procedure closure is a pair <I,A> where I is a procedure image and A is an association mapping the global symbols of I, and no others."

UK-37 Accepted. Reword sentence to "A *complete* procedure closure is a procedure closure for which all the global symbols of the image are mapped."

UK-38 Accepted. Sentence starts with "A *partial* procedure closure is a procedure closure for which ...."

UK-39 Accepted

UK-40 Yes. In general all the defining occurrences should be italicised.

UK-41 Accepted.

UK-42 Accepted.

UK-43 IAssoc and GIAssoc are used to define the transitive closures.

The following two paragraphs should be included in section 6.13 (i.e. before 6.13.1):

In order to be able to discuss the set of global variables shared by two procedures, or to define pointer (or parameter) aliasing, it is necessary to know when one value is "referenced by" or is "accessible from" another value. X is a <<italic>> simple associate <<roman>> of Y if X can be obtained from Y by following pointers or extracting the elements of aggregate values. X is a <<italic>> generalized associate <<roman>> of Y if X can be obtained from Y by combinations of the above actions and by invocating procedures.

The next two clauses formalize these two concepts.

Replace the 1st paragraph of 6.13.1 by:

The concept of simple associates is embodied in two functions.

Replace the 1st sentence of 6.13.2 by the following two paragraphs:

The concept of generalized associates includes values that can be obtained with the help of other procedures. Again, two functions are needed.

Generalized Immediate Associates is defined as follows:

In 6.14, replaces the 2 uses of "the Association(" with "the Assoc(".

UK-44 6.15, 5th paragraph. Strike "forward and backward".

UK-45 Rejected for "sharing structure": normal English usage. Replace ""congruent" to" by "as in". Replace "For example," by "An example is".

UK-46 Top of page 24. Answer: YES.

UK-47 In 6.16, 1st sentence, replace "complex" by "composite".

Put the next paragraph after the 1st paragraph:

The following definitions are not used in this International Standard, but will help shorten definitions in binding standards.

Put the following parts of the definitions in italics:
T <<italic>>is an identity on<<roman>> datatype Q ...
T <<italic>>maps<<roman>> F <<italic>>to<<roman>> F'
T <<italic>>maps<<roman>> Q <<italic>>into<<roman>> Q
T <<italic>>preserves<<roman>> datatype Q

Replace the paragraph starting "TF" and the following text with the following note:

Note - As an example of how the above concepts can be used, assume that we need to define a translation procedure TF that replaces all boxes in a value V with new ones while preserving the sharing structure within V.

TF(V) is computed as follows:
(1) <<existing text>>
(2) <<existing text>>
(3) Let Z be a function Z: Value -> Value satisfying
<<existing text>>
(4) Finally set TF(V) = Z(V)

UK-48 Take text for added-character from ISO 11404

UK-49 Accepted

UK-50 Accepted

UK-51 No. However on the issue of the consistency of Annex c: care should be taken that the IDN in LIPC is a real superset of the LID IDN.

UK-52 Accepted.

UK-53 1- within the IDN, two definitions who only differ in case are conflicting.
2- all reference uses of an identifier must agree with the defining use in case of letters. Consequently all uses of an identifier with an IDN text are consistent in case, and a binding which preserves case will be suitable for case sensitive and case insensitive languages.

NZ-1 Accepted

NZ-2 Accepted

NZ-3 Accepted

NZ-4 Accepted, but proposed text is wrong. Definition should be: "A parameter is used to communicate a value from a client to a server procedure. The value supplied by the client is the actual parameter, the formal parameter is used to identify to received value in the server procedure."

NZ-5 Accepted. Note that the terms global and local are used with respect to the duration of bindings, and not with respect to language scoping rules.

Add after 1st sentence of 3.32: Procedure closures may be complete, with all global symbols mapped, or partial with one or more global symbols not mapped.

Add definition of global symbol; take text from 6.3.

NZ-6 Accepted in principle. The global state might include not just a program, but a whole execution environment. See response on NZ-9.

Replace "series" by "succession".

NZ-7 Accepted. Take new definition.

NZ-8 Rejected. This definition will be rephrased as soon as the RPC situation is cleared-up.

NZ-9 It has been clarified by the change in 3.5: a box is no more abstract than a program variable. We do not consider any piece of the mathematics as defined in this document, more abstract than any other.

NZ-10 Accepted. The definition will be aligned with TR 10176. Same for 3.15.

NZ-11 These concepts are given complete mathematical definition in the document. The definition cannot be read without reading the full model.

NZ-12 The definition of interface closure was defined as is to be explicit. The fact that some information in the definition may be redundant was considered to be not important.

NZ-13 Rewording along the lines suggested for 3.18 is rejected: see response to NZ-12.

3.19 is an incorrect duplicate of 3.24, and should be removed.

NZ-14 Accepted. Remove "(A)". Last sentence: "binding to values".

NZ-15 Accepted.

NZ-16 Accepted in principle; remove definition of name, as it is used only in the ‘normal’ sense.

NZ-17 Accepted in principle; remove definition of program text as program text is not the topic of discussion, and is used only in the normal sense. Check text for occurences of program text.

NZ-18 Accepted; Rephrase as: A program entity used to refer to a value.

NZ-19 Rejected. The term ‘written’ is used in its colloquial form, which can include ‘written by automata’.

NZ-20 Accepted.

NZ-21 Accepted. Reword to "Therefore an actual parameter is any value of the datatype required by the call."

NZ-22 Accepted; Reword to "as in the previous case".

NZ-23 Accepted. See resolution of UK-27.

NZ-24 Accepted. See resolution of UK-30.

NZ-25 See NZ-6. Remove ", modelled ... sequence.". In 2nd sentence, replace "series" by "succession". Add as last sentence of the paragarph: "An instant in time corresponds to a state in an execution sequence."

NZ-26 Is already rephrased.

NZ-27 Accepted. See UK-35.

NZ-28 Accepted. Should be sub-clause. Check the document for section.

NZ-29 Accepted.

NZ-30 Accepted.

NZ-31 Accepted. This will be copied from LID when LID stabalizes. Why is the "low line" in LID (table 7-1) called hyphen??

NZ-32 Acceptec. All references will be to "ISO/IEC 10646-1:1993".

NZ-33 Accepted. See NZ-31.

NZ-33 See UK 52.

NL-1 Accepted. Add sentence at end of paragraph 1 of 5.2.1.2: "This can be done at the beginning of the call, or while the call is in progress."

Note 2: An example of the use of Call by Value Send on Request is when a client wishes the server procedure to record a time, and whishes that to be done at a specific point during the execution of the call, rather than at the initiation of the call.

Note 3: replace "the value supplied" by "that same value". Start of note: "The use of the server of a parameter of the Call by Value Sent on Request type can be .."

WFW-1 Issue: 7.2.1 1st sentence I assume that the identifier is used in the interface-body, and is not the identifier from the interface-identifier. Reword to "If an identifier in a type definition in an interface-body is used to refer to ..."

Solution: Start with: "In an interface body, an identifier used to refer ..."

WFW-2 Issue: 7.2.2 1st sentence: See WFW-1

Solution: Same rephrasing as in WFW-1

WFW-3 Issue: 7.4 Declarations
Does this one-liner merits a full section on its own? And if so, should not 7.5 be 7.4.1 And where are the procedure-decl and termination-decl defined? Should the procedure-dcl not be the procedure-type, which is already included in type-decl? Should the title of 7.6 not be Type Declarations?

Solution: Make line from 7.4 as 5th rule of 7.2, remove 7.4. Title of 7.3 "Import Declarations". In 7.2 the word interface has no quotes around it. Check references (termination-reference is defined twice in the annex)

WFW-4 Issue: 7.5 Value Declarations
Move the rule of value-expr immediately after the rule for constant-type-spec. Why is the enumeration-literal missing from value-expr? The 3rd paragraph (An interface shall only define

constants of ...) is a restriction on the use of the IDN. Is this explained anywhere?

Solution: OK for the move. Add enumerated-literal. Remove 3rd paragraph. (remainder from RPC)

WFW-5 Issue: 7.6.1.7 The procedure datatype
As the difference between a function and a procedure is the presence of the "returns" "(" return-argument ")", it is more convenient in a binding to have this as one non-terminal.

Solution: Rejected

WFW-6 Issue: 7.6.1.7.2 Procedure values, 7.6.1.7.3 Subtypes
This text is also in 11404. Should it remain here, or just a reference? In any case, most of it has hardly anything to do with the IDN.

Solution: Leave.