ISO / IEC JTC1 / SC22
Programming languages, their environments and system software interfaces
Secretariat: CANADA (SCC)

ISO/IEC JTC1/SC22
N 1347

APRIL 1993

TITLE: Summary of Voting and Comments Received on the CD Registration ballot for Working Draft on: Language-independent Procedure Calling Mechanism

SOURCE: Secretariat ISO/IEC JTC1/SC22

WORK ITEM: JTC1.22.16

STATUS: New

CROSS REFERENCE: N1289

DOCUMENT TYPE: Summary of Voting - CD registration

ACTION: For information to SC22 Member Bodies. See attached.

Address reply to: ISO/IEC JTC1/SC22 Secretariat
J.L. Côté
Treasury Board Secretariat
140 O'Connor St., 10th Floor, Ottawa, Ontario, Canada, K1A 0R5
Tel.: (613)957-2496 Telex: 053-3336 Fax: (613)957-8700

SUMMARY OF VOTING ON:

Letter Ballot Reference No: SC22 N1289
Circulated by :JTC1/SC22
Circulation Date :1992-12-17
Closing Date :1993-03-19

SUBJECT: CD Registration ballot for: Working Draft on Language-independent Procedure Calling Mechanism

---

The following responses have been received:

'P' Members supporting CD registration, without comments : 06

'P' Members supporting CD registration, with comments : 03

'P' Members not supporting the CD registration : 00

'P' Members abstaining : 02

'P' Members not voting : 10

Secretariat Action:

The Secretariat will forward the attached comments to WG11 for consideration. Upon receiving recommendation from WG11, N1289 or a revised version will be registered as Committee Draft.

ISO/IEC JTC1/SC22 LETTER BALLOT SUMMARY

PROJECT NO: JTC1.22.16

SUBJECT: CD Registration ballot for: Working Draft on Common Language-independent Procedure Calling

Reference Document No: N1289
Circulation Date: 1992-12-17

Ballot Document No: N1289
Closing Date:1993-03-19

Circulated To: SC22 P, L

Circulated By: Secretariat

SUMMARY OF VOTING AND COMMENTS RECEIVED

	Approve	Disapprove	Abstain	Comments	Not Voting
'P' Members					
Australia	( )	( )	( )	( )	( )
Austria	( )	( )	( )	( )	( )
Belgium	(x)	( )	( )	( )	( )
Brazil	(x)	( )	( )	( )	( )
Bulgaria	( )	( )	( )	( )	( )
Canada	( )	( )	(x)	( )	( )
China	( )	( )	( )	( )	( )
Czechoslovakia	( )	( )	( )	( )	( )
Denmark	(x)	( )	( )	( )	( )
Finland	(x)	( )	( )	( )	( )
France	( )	( )	( )	( )	( )
Germany	(x)	( )	( )	( )	( )
Greece	( )	( )	( )	( )	( )
Italy	( )	( )	(x)	( )	( )
Japan	( )	( )	( )	( )	( )
Netherlands	(x)	( )	( )	( )	( )
New Zealand	(x)	( )	( )	(x)	( )
Romania	( )	( )	( )	( )	( )
Sweden	( )	( )	( )	( )	( )
Switzerland	( )	( )	( )	( )	( )
UK	(x)	( )	( )	(x)	( )
USA	(x)	( )	( )	(x)	( )
USSR	( )	( )	( )	( )	( )
'O' Members					
Argentina	( )	( )	( )	( )	( )
Cuba	( )	( )	( )	( )	( )
Hungary	( )	( )	( )	( )	( )
Iceland	( )	( )	( )	( )	( )
India	( )	( )	( )	( )	( )
Poland	( )	( )	( )	( )	( )
Portugal	( )	( )	( )	( )	( )
Singapore	( )	( )	( )	( )	( )
Turkey	( )	( )	( )	( )	( )
Thailand	( )	( )	( )	( )	( )
Yugoslavia	( )	( )	( )	( )	( )

New Zealand Comments on JTC1/SC22/N1289

p4, 4.1

What character set is to be used for writing the IDN syntax? Are the characters visible throughout the syntax as terminal symbols to be treated as the (short) names of characters in some encoding (eg ISO 10646)? This needs to be made clear.

p10-11, 6.13

The development of a generalised version of *spec* makes quite confusing reading in a (draft) standard. It is suggested that such mathematical development should be relegated to an informative annex or, if there is insufficient to make it worth generating a separate annex, place it in an informative note.

As an additional comment on this, however, in the third version (second on p11) there are terms P which appear to be completely undefined. Please define them fully.

p11,6.14

If, should a signal occur, the W_j are to be of "type FT_j" then this *must* be in the closure specification! Unfortunately, however, this would seem to violate the type correctness unless FT_j and RT_j are everywhere identical types. This raises the immediate question that the mechanism modelled may need to explicitly postulate the existence of the value *undefined* in every type domain [at least for the purposes of the LIPC] so that indeed FT_j and RT_j are identical but that some elements of the result sequence may be *undefined*.

This matter raises a further interesting side issue that an alternative model which may read rather better would treat the results as a set rather than a sequence. The latter term implies some order of result production which is, of course, meaningless. The former model, however, could permit a signalled termination to occur with some results not 'in' the set of results (a possible alternative to having to provide an explicit *undefined* value in each type domain).

p15, Example

Please use formal impersonal English in explaining this.

p22, 7.1.5.1.3

This refers to ISO 2375, ISO 7350 and ISO 10646 – but none of these appear in the Reference in Section 2, p2.

p30, 7.1.10

The second note is truly irrelevant. The interface definition *must* specify, by indicating all possible synonyms, the identity of an object using the names for characters specified in the character type definition (see p22). If a name may be either "fred", "FRED", "fRed", ... then this is a necessary part of the specification and the standard *must* require appropriate mappings by source and destination translators using common translation rules. Where dynamic binding may be involved different translation rules may be appropriate even within one computer system at one time (as an aside it will almost inevitably reflect on the SC21 work in CD 11578!).

UK COMMENTS ON SC 22 N 1289:
Language Independent Procedure Calling

The UK votes YES with the following comments:

1. The datatype definitions in LIPC appear to be mainly duplicating those in LID:

   The definitions of datatypes should be by reference to LID, together with any necessary amendments, with a summary of the appropriate sections of LID provided as an informative Annex.

2. There are some inconsistencies in the IDN between LIPC and LID:

   It is still the UK position that the IDNs in the LIPC, LID and the RPC standards be aligned at the final IS stage.

3. "Global IDN" does not appear to be defined:

   This term should be defined, furthermore there should be an overall description of this and the relation between LID, LIPC and RPC provided as an Informative Annex to all three standards.

4. It is not clear how the binding to another programming language is achieved:

   The next version of this document should include as an informative annex binding with at lease two languages which are significantly different in procedure calling concepts.

5. Attention should be given to ensure that normative text and explanatory comment are clearly distinguished.

The UK believes that these points must be addressed before a document is proposed for DIS registration. Progress in these areas will determine its vote at that point.

JMS/JMK
11 March 1993

X3T2/93-022

U.S. Position on LIPC CD Registration Ballot
January 8, 1993

The US votes YES on the registration of LIPC working draft 6 as a Committee Draft with the following comments:

Change "abnormal" to "non-normal" throughout document.

In clause 6, all [-->] should be [->].

Add definition of "program text" to clause 3. The form in which a program is submitted to a language processor.

Add definition of "partial procedure closure" and "complete procedure closure" to clause 3. See the text in clause 6.6.

Add definition of "name" to clause 3. An identifier or an identifier qualified so as to make it unique within some scope.

Replace all occurrence of "run" with "execute" throughout the LIPC.

- i.e. run-time --> execution-time and other variants

Change last sentence of clause 6.1 "Value will contain boxes and procedure closures as described in clause 6.8.

Change the title of clause 6.2 to Boxes and Global State

Replace the two paragraphs following the first note in clause 6.2 with the following:

"The global state is the set of all existing boxes and their currently assigned values. It is the unique characteristic of boxes that their operations involve the global state: Read accesses the global state; Create and write produce a new global state."

Remove the first note from clause 6.2.

Add note at end of clause 6.2 : "In cases of concurrent processing, the series of global states making up the execution sequence can not necessarily be determined by examining the program text and may vary from execution to execution."

Remove the last sentence of clause 6.2, "All boxes are..."

Change the first sentence of clause 6.3 to "A symbol is a reference in a program text to a value (including boxes and procedure closures)."

Change the second sentence of clause 6.3 to "These referenced values are the values that the program can access directly during execution."

Change the bullet list in clause 6.3 to:

- Global symbols are used to refer to values that exist prior to invocation.

- Local symbols are used to refer to values that are created at invocation (e.g., the local "stack frame" variables).

- Argument symbols are used to refer to values that are the arguments of a particular invocation.

Change sentence in clause 6.3 before the bullet list to start "The symbols of a particular procedure fall into three disjoint categories:"

Add a note at end of clause 6.3 "Local symbols and argument symbols of one procedure may be global symbols of another procedure (e.g., nested procedures).

Add a note 2 at end of clause 6.3 "How identifiers used in a particular programming language relate to references to values is defined by the scoping rules of the language. In particular the same identifier may relate to different references in different contexts and thus would be two different symbols within the subclause."

Change the second sentence of clause 6.4 to "is the procedure type... and the list of..."

In clause 6.4 add a note after the operation definitions: "The ordering within the sequence produced by gsyms and lsyms is seldom relevant, however the ordering within the sequence produced by asyms is important (see clause 6.7)."

In clause 6.4 first paragraph after operation change "argument symbols" to "input argument symbols".

In clause 6.4 after first paragraph after operation add "Note: Some language processors output arguments will appear in the local symbols list. Further, some languages may also have all of the input argument symbols appear in the local symbols list, where the language actually makes copies of the arguments and these copies are actually manipulated during the execution. Both of these are language issues that do not affect the LIPC model."

Change last paragraph of clause 6.4 to a note "Procedure images are created by the language processor. How a procedure image is created is outside the scope of this draft International Standard."

In clause 6.5 change [x -> y] to "denotes the association that maps the symbol x to the value y and maps no other symbols"

In clause 6.5 change "dom" and "rng" to "domain" and "range". Check for other occurrences.

In clause 6.6 remove the last sentence of the last paragraph.

In clause 6.6 change the last paragraph to be a note similar the the note at the end of clause 6.4.

In clause 6.6 end of second paragraph and a forward reference to clause (6.xx) detailing partial procedure closures. A new clause 6.xx needs to be created.

In note in clause 6.6 change "activated" to "invoked". Change "activation" to "invocation".

In clause 6.6 define complete procedure closure - a closure where all of the global symbols are mapped. Add this sentence before paragraph two.

In clause 6.6 change the old paragraph two "A partial procedure closure is a closure where at least one of the global symbols is not mapped."

In clause 6.7 change first paragraph to "Basic invocation is an operation on complete procedure closures described as follows:"

In clause 6.7 change <I,A> to <image, assoc>. Change I and A throughout section to image and assoc (not the subscripted A's).

In clause 6.7 sentence before bullets change to "During execution a series of changes may be made to the global state. When and if execution terminates a value in Status x Sequence(Value) will result, and the association Q is lost." Remove the bullets and the sentence following the bullets.

In clause 6.7 change the last sentence of the last paragraph to a note and some word smithing will likely be necessary.

In clause 6.7 first paragraph fourth sentence, change to "The second sequence of values represents the values resulting from the invocation."

In clause 6.7 after "LBi create()" add "Define invocation association Q by:" Delete "Let" from next line.

In clause 6.7 add a note after Let Q, "Note: This implies for those processors which copy input arguments using local symbols as mentioned in clause 6.4 the local symbol associations in Q will take precedent over the argument symbol associations."

Change 6.8 to "Relationship to LID" Delete old text of 6.8.

Add a "6.8.1 Pointers and Boxes"

Every box is a value of some pointer datatype.

The element type of the pointer type is the datatype of the value IN the box as distinct from the box itself which is the value of the pointer type.

A value of type T can be assigned to a box whose value is of type pointer to T.

The read operation is the LID dereference operation.

The create and write operations of clause 6.2 are additional operations on boxes not discussed in LID.

Add a "6.8.2 Procedure Values"

The concept of procedure closure (see 6.6) is the appropriate extension of the LID concept of procedure value to include the concept of global state. Thus a procedure closure is regarded as a value of some procedure type.

In clause 6.9, delete old text and add new stuff below

The notion of instances of values common to programming languages is modeled here with boxes and values in boxes.

A instance of a value is really a box containing such a value.

Note: In a real computational environment boxes have the concept "location". LIPC does not attempt to model location of boxes, but adding location is recognized as a compatible extension.

When a box is passed from one entity to another both of those entities have access to the same box and see all modifications to that box. This is true even when these two accessors are at two different "locations". When two accessors at two different locations access this box they are accessing the same conceptual box, i.e. the same element of the global state.

Note: Some mechanism cannot support sharing and thus cannot actually pass boxes but are forced to create new boxes. Therefore the relationship between boxes and locations is a procedure calling mechanism problem.

Delete clause 6.10 Pointers and renumber appropriately.

In clause 6.11, rewrite first paragraph and make appropriate change to clause 3 using the first sentence as the new definition, "An interface closure contains a collection of procedure closures and a collection of names with a mapping between them. This is modelled as an association that maps the set of names to procedure closures."

In clause 6.12, rewrite first paragraph and make appropriate change to clause 3 using the first sentence as the new definition, "An interface type contains a collection of procedure closures and a collection of types with a mapping between them. The mathematical representation of this is an association that maps a set of types to procedure closures."

In clause 6.12 by the example delete "X,Y, and Z are procedure closures,".

In clause 6.14, copy two lines after Let spec(C) example to the IDN section.

In clause 6.14, change IN and INOUT to "input" and OUT and INOUT to "result" and delete "plus...". Add to the end of the sentence "where E1 through Een are the non-normal terminations.

In clause 6.14 add an "and" between n=an and For.

In clause 6.14 move "For..." after "Vi is a..."

In clause 6.14 change "If invocation of C" to "If and when the invocation of C"

In clause 6.14 replace sentence "If C is written..." by "The termination is type correct if"

Move clause 6.7 Basic procedure invocation before clause 6.14 Type correctness

In clause 6.15 break this section into 6.15.1 Simple Associates and 6.15.2 Generalized Associates. Break at "To include such...".

In clause 6.15, Add "IAssoc(x) is defined as follows:" break the Ifs into 4 bullets. Change IAssoc(x) = {v} to the set consisting of the single value v. Change IAssoc(x) = {} to the empty set.

In clause 6.15 change the notation "union" to a big U.

In clause 6.15 after "range(a) U {v1..." change "(with some computation)" to "(with some computation not including procedure invocation)"

In clause 6.15 delete the paragraph beginning "The invocation of <I,A>..." and the note following this paragraph.

In clause 6.15 last paragraph change "potentially write...them as well." to

"access any value in GZ."

In clause 6.15 replace last sentence of last paragraph start with "If y is a" by "The only elements of the global state that the invocation of <I,A> on <v1,...Vn> can access or modify are those which are boxes in GZ

In clause 6.15 the last note, change bullet 3 to "J is in range(Q) and B can be constructed out of accessible values."

Delete clause 6.18 Execution Context. Add a new clause after 6.15 Associates as 6.16 Execution and invocation contexts.
The execution context of the procedure closure <I,A> is the set of all boxes in the Assoc (range(A)).
The invocation context of a particular invocation of the procedure closure <I,A> is the set of all boxes in the Assoc (range(Q)) where Q is the invocation association of this invocation of <I,A>.
Note: Both the execution context and the invocation context can vary over time during the execution of I, due to assignment of boxes into the old context.

In clause 6.19 at the end add "In any conforming implementation any of the marshalling interfaces, providers, or data translation into a transmittable form may actually be collapsed into combined steps or even no steps. The limit of which is the client procedure directly calling the server procedure."

Move clause 6.19 to clause 6.1. Move 8 except 8.7 and 8.8(including 8.2 and 8.4) up with clause 6.1.

In clause 8.2 change line two "but only" to "each". Delete the last paragraph of clause 8.2.

Combine clauses 8.2, 8.4, and 8.8 and place after clause 8.3. The 8.2 and 8.4 are subsidiaries of the 8.8 notion.

Rename clause 8.1 to Sent on Initiation and clause 8.3 to Return on Termination.

Move the contents of clause 8.5 to clause 8.

The notion aliasing needs to be added to LIPC, and the aliasing notion in RPC is not the suggested addition.