From: Frank Farance
Organization: Farance Inc.
Telephone: +1 212 486 4700
Fax: +1 212 759 1605
E-mail: frank@farance.com
Date: 1995-08-25
Document Number: WG14/N462 X3J11/95-063
Subject: X3J11 Review of ISO DIS 13886

Review of: ISO DIS 13886, ``Information technology -- Language-independent procedure calling''

Please respond to me via E-mail if you have any further questions.

Thank you for asking us to review your document. The following are suggestions from various technical experts in ANSI X3J11. The comments are organized as procedural, general, and specific comments. X3J11 recommends the US TAG (X3T2) vote ``NO'' on this DIS.

PROCEDURAL ISSUES

The cover sheet specifies a 4-day review period for developing the position of the US TAG (from 1995-07-10 to 1995-07-14). This is an unreasonable review period.

GENERAL ISSUES

Overall, the LIPC model seems well thought out and applicable to many procedure calling mechanisms.

The use of mathematical notation for *this* specification is suboptimal. The notation used for boxes, associations, etc., is somewhat confusing because this isn't conventional mathematical notation. Another problem with the notation is that it is difficult to measure conformance to this standard.

Both a lightweight (e.g., same address space) and heavyweight (e.g., RPC-like) procedure calling can be envisioned with LIPC. It is not clear how both would co-exist in the same language. For example, choosing a lightwieght-only implementation would provide the typical subroutine calling features, but would handle the contingencies for RPC-like calls. Choosing a heavywieght-only implementation would be much more general, but would probably have too much overhead for same address space calls.

The caller and called procedures would need to negotiate features (e.g., assertion-oriented, inquiry-oriented, etc.). Some of the features to be negotiated are:

- The error handling capabilities.

- The multi-tasking capabilities.
- The interrupt handling capabilities.

For example, procedure A calls B. If A and B are in separate processes or different machines, this would work without problem. But if A and B are in the same process, if A calling B might cause the process to deadlock if an RPC-like mechanism was used and there was no multi-threading.

The use of LID may cause problems with LIPC. While it is desirable to have a generic type, such as integer, the implementation issues (e.g., precision, bit/byte ordering/alignment, data representation) are significant aspects of the interface. How will LIPC address these in its interface definition language?

SPECIFIC ISSUES

The following terms are either undefined or have ambiguous meanings:

- characteristic function
- procedure closure
- virtual contract
- actual contract
- closed sequence

A sample binding should be included for each of the languages as was done for LIA-1.

The "wrap" procedure addresses only one aspect of ''thunking'' procedures. Another approach would be caller-only or called-only interfacing -- important for accessing legacy code.