The minutes of the 1980 Fortran Experts Meeting were produced in a style different from the form adopted later. This file contains the following separate documents: Minutes, Agenda, Attendance List, Documents distributed at the Meeting and Summary of Discussions. Reports 56 and 59 (below) have been reproduced as WG5-N025. Other papers referenced from the text are in X3J3 document X3J3/133.

This file was compiled in April 2010 from the original paper versions using OCR and retaining the same style of presentation.

Fortran Experts Group
Meeting at Amsterdam, 20 - 23 October 1980

MINUTES

1. Opening

The meeting was opened at 10:30 am on Monday October 20 by the Convenor, Jeanne Adams. The Host, H J Bok of Nederlands Normalisatie-Instituit, welcomed the delegates. Jeanne Adams was elected Chairman for the meeting, and Loren Meissner was appointed Secretary for the meeting.

Delegates were present from the following countries: Austria, Canada, France, Germany, Hungary, Netherlands, Sweden, United Kingdom, and United States. Delegates from ECMA and EWICS were also present.

The Agenda shown in Document 89 was adopted.

Reports were presented by delegates from each of the countries, and from ECMA and EWICS.

2. Technical Discussions

The following topics were discussed:

Summary of X3J3 proposals for the next revision of Fortran.

United Kingdom comments on X3J3 proposals

Standardizing compiler listings and diagnostic messages.

Control structures.

Internal procedures.

Global data sharing.

Fortran source form, significant blanks, and relational operators.

Language architecture.

Numerical precision.

Generalized Real data type.

Interface solution to Application Facility modules.

Conformity module.

EWICS TC1 activities toward the standardization of Industrial Real-time Fortran.

Implementation of Real-time Fortran as Application Module or Environment Module.

Users' view of Real-time Fortran.

Fortran 77.

Generalization of Fortran.

General or specific solution to multitasking semantics.

Most of these topics are covered in Position Papers listed in the Table of Contents. Discussions are summarized in Document 90.

3. Closing

Requirements concerning a subsequent meeting were discussed.

In connection with the SC5 meeting in London during October 1981, there will be some opportunities for persons interested in Fortran to meet and discuss mutual concerns. However, there will not be a full "Fortran Experts Group" meeting at that time.

The delegates decided to accept an invitation from Austria to hold the next meeting of the Fortran Experts Group in Vienna. A preference was expressed for holding the next meeting of this Group during June 1982, rather than Fall 1981. Austria will communicate with the Convenor, Jeanne Adams, to confirm the exact time and meeting place for an invitation to Vienna during June 1982.

The delegates expressed appreciation to the Host, H J Bok of Nederlands Normalisatie-Instituut, and to Kees Ampt and other members of the NNI Fortran Group who assisted with local arrangements.

The meeting was adjourned at 5:30 pm on Thursday 23 October.

american national standards committees
X3-computers and information processing
X4--office machines and supplies
operating under the procedures of the
American National Standards Institute

milestone

reply to

Jeanne C. Adams
Computing Facility
National Center for Atmospheric Research
P.O. Box 3000, Boulder, Colorado 80307
(303) 494-5151 FTS-322-5523

25 September 1980

AGENDA

ISO/TC97/SC5 FORTRAN Experts' Meeting

Amsterdam,
The Netherlands October 20-23, 1980

1. Opening of the meeting (10:30 a.m.; Jeanne Adams, Convener)

2. Welcome of delegates

3. Election of chair

4. Roll call of delegates

5. Adoption of agenda

6. National activity reports

7. Procedural matters

8. Technical reports

   - Monday, October 20, 2:00 p.m.

     Summary of X3J3 Proposals for the next revision of FORTRAN
     (Jeanne Martin and Jim Matheny, X3J3, U.S.)

     Discussion
     (Martin Greenfield, X3J3, U.S.)

- Tuesday, October 21, 9:00 a.m. to 12:30 p.m.

  Numerical Precision
  (J. Reid, Harwell, U.K.)

  Discussion
  (D. Muxworthy, U.K.)

  Generalized Real Data Type
  (J. Lawrie Schonfelder, U.K.)

  Discussion
  (Alan Wilson, ICL, U.K. and X3J3, U.S.)

- Wednesday, October 22, 9:00 a.m.

  Interface Solution to Application Facility Modules
  (Jerrold Wagener, X3J3, U.S.)

  Discussion
  (Loren Meissner, X3J3, U.S.)

  Completeness Module
  (A. Clarke, Rothamsted, U.K.)

  Discussion
  (Murray Freeman, X3J3, U.S.)

- Wednesday, October 22, 2:00 p.m.

  EWICS TC1 Activities toward the Standardization of Industrial Real Time FORTRAN
  (Wilfried Kneis, Chair EWICS TC1, Germany)

  Discussion
  (Kees Ampt, Chair FORTRAN Group, NNI, The Netherlands, and
  Richard Signor, Chair, Ad hoc Task Group X3J3.2, U.S.)

  Implementation of Real Time FORTRAN 198x as Application Module or Environment Module
  (Jan Snoek, Chair, Working Group Real Time FORTRAN, NNI)

  Users' View of Real Time FORTRAN
  (Zom Dannigs, Phillips, The Netherlands)

- Thursday, October 27, 9:00 p.m.

  Discussion of FORTRAN77
  (Betty Holberton, Bureau of Standards, U.S.)

  Generalization of FORTRAN
  (Ingmar Dahlstrand, Sweden)

General or Specific Solution to Multi-Tasking Semantics
(Kees Ampt, Chair, FORTRAN Group, NNI, The Netherlands)

9. Action on recommendations

10. Requirements concerning a subsequent meeting

11. Any other business

12. Adjournment

Fortran Experts Group
Meeting at Amsterdam, 20 - 23 October 1980

ATTENDANCE LIST

Jeanne C Adams, Convenor and Chairman
H J Bok, Host
Loren P Meissner, Secretary

1 Jeanne C Adams, United States
2 Cornelis G F Ampt, Netherlands and EWICS
3 Anton Bickle, Canada
4 Pierre Boutrouille, France and ECMA
5 P Alan Clarke, United Kingdom
6 Ingemar Dahlstrand, Sweden
7 Martin N Greenfield, United States
8 Frances E Holberton, United States
9 Kalman Janko, Hungary
10 Wilfried Kneis, Germany and EWICS
11 A Kranendonk, Netherlands
12 Jean Louis Lacour, France and ECMA
13 Luigi Lauri, ECMA
14 Karlotto Mangold, Germany, ECMA and EWICS
15 Jeanne T Martin, United States
16 Christian J Mas France and ECMA
17 James H Matheny, United States
18 Loren P Meissner, United States
19 Michael Metcalf, United Kingdom
20 Meinolf Munchhausen, Germany and ECMA
21 David T Muxworthy, United Kingdom
22 Peter Penders, Netherlands
23 John Reid, United Kingdom
24 E H Reitsma, Netherlands
25 Karl-Heinz Rotthauser, Germany
26 Gerhard J Schmitt, Austria
27 J Laurie Schonfelder, United Kingdom
28 M K Shen, Germany
29 Richard W Signor, United States
30 Jan A M Snoek, Netherlands and EWICS
31 Hieronymus Sobiesiak, Germany and EWICS
32 Johan W van Wingen, Netherlands
33 Nico Vossenstijn, Netherlands
34 Jerrold L Wagener, United States
35 Alan Wilson, United Kingdom and United States

Fortran Experts Group
Meeting at Amsterdam, 20 - 23 October 1980

DOCUMENTS DISTRIBUTED AT MEETING

56 Report of Fortran Experts Group Meeting, November 1979

59 (Part 3; formerly Doc. 59) Report of Discussions, November 1979

60 Austria: Invitation for Fortran Experts Group Meeting, 1981

61 United Kingdom: National Activity Report

62 United Kingdom: Comments on X3J3 Proposals

63 D T Muxworthy: Standardizing Compiler Listings and Diagnostic Messages

64 P A Clarke: Control Structures

65 J L Wagener: Fortran Module Interface Mechanisms

66 W Kneis and H Sobiesiak: EWICS TC1 Activities towards the Standardization of Industrial Real Time Fortran

67 J T Martin and J H Matheny: Summary of X3J3, Proposals for the Next Revision of Fortran

68 P A Clarke: Conformity Module

69 J K Reid: Language Requirements for Numerical Software

70 J L Humar: Internal Procedures

71 J L Humar: Global Data Sharing

72 Z Miller: Fortran Source Form, Significant Blanks, and Relational Operators

73 A Bickle: Language Architecture

74 Canada: National Activity Report

75 Netherlands: National Activity Report

76 Sweden: Position Papers on Fortran 8x

77 List of Delegates (Preliminary)

78 United States: National Activity Report

79 J C Adams: Exercises on Core and Modules

80 J L Schonfelder: A Proposal for a Generalised Real Data Type

81 J L Schonfelder: Generalised Real Data Type

82 K W Hirchert: Proposals -- Groups and Internal Procedures

83 J Snoek: Implementation of Real-time Fortran in the 198x Standard as Application Module or Environment Module?

84 T Dannijs: Users View of Real Time Fortran

85 List of Documents

86 Attendance List

87 ALGOL -- Second Draft Proposal

88 C G F Ampt: General or Specific Solution to Multi-Tasking Semantics?

89 Agenda, October 1980

[The paper numbering scheme then used by the Experts' Group was: 1978:3-36, 1979: 37-59, 1980: 60-91. Summary of Discussions was 90, Minutes was 91. The full set of papers is in X3J3/133.]

Fortran Experts Group
Meeting at Amsterdam, 20 - 23 October 1980

SUMMARY OF DISCUSSIONS

1. Opening Remarks by Chairman.
Review of the history of ISO participation in the standardization of Fortran, beginning in 1972.

2. National Activity Reports.

See Documents 61, 74, 75, and 78.

3. Summary of X3J3 Proposals for the next Revision of Fortran, by Martin and Matheny.

See Document 67.

Module and Prefix Rules
Dahlstrand: If there is a hierarchy in the use of modules, the first module could contain the Intrinsic Functions. Then a later statement could remove their names from "reserved" status.

Array Intrinsics
Ampt: Are the values actually produced?
Dahlstrand: When "all scalar operations" are extended to arrays, this should include operations defined by the user.

Groups and Internal Procedures
Reid: Is a Group a compilation unit?

A: That is what we had in mind.

Can Internal Procedures be nested?

A: Not settled. Probably not.

Global Data
Schmitt: What can be in the list? Symbolic constants?

A: Not sure about Symbolic constants. More than just variables, e.g., Procedures, data structure Forms.

Character extensions
Schonfelder: "Same As" passed character length?

A: Not contemplated.

4. UK Comments on X3J3 Proposals, by Muxworthy.

See Document 62.

The general feeling is that X3J3 is going "too far, too fast". There will be considerable Fortran 77 experience by 1983-1984, so the pace of X3J3 activities must be slow enough to build on this experience. When converting to Fortran 77, people may avoid using features slated for deletion.

[In answer, it was pointed out that according to the present X3J3 time table, final ANSI completion is scheduled for 1985. If, as with Fortran 77, compilers appear 3 years later, the new Fortran would not be in wide use before 1988. The Obsolete Features Module will be part of the Fortran Standard until almost the end of the century.]

Concerns

Irregularities in Fortran 77

Fortran 77 deficiencies not being addressed by X3J3 (e.g., Namelist?)

Matrix inverse -- can it be done well?

Multi-level exits are needed.

Dependent compilation, as implied by Groups.

Interfaces to other languages -- what is needed?

ISO character sets -- what about control characters?

Long names, external.

Too many asterisks.

Variables in format?

Too much freedom in E output, makes checking difficult.

Conversion between external and internal numeric forms should be the same for the compiler as for the input system. [But "cross-compiling" would be difficult if this were an absolute requirement.].

Discussion

Schonfelder: We like each specific new feature, but they don't always fit together as well as they should.

Bickle: We would like a less extensive revision, that would be completed sooner.

5. Control Structures, by P A Clarke.

See Document 64.

Multi-level Exits

Schmitt: Difficulties can be caused by using "Alternate Returns". Consider having a special data type (not integer or asterisk) for labels, that could be passed as arguments.

A: Multi-level exit is better than passing a decision back one level at a tine through several levels.

Ampt: Clarke's proposal seems to be a disguised form of Alternate Return.

6. General Discussion, led by Greenfield.

Metcalf: Bit Data Type is very useful to High Energy Physics; should he in core.

A: We do not yet know what will be in core.

Metcalf: Source form: Worried about effect on readability -- statement label inside a line seems bad.

A: You don't need to use it. Listing can decompose it. There are arguments for and against this. There is no difficulty from a syntactic standpoint.

Ampt: Significant blanks helps.

To keep the spirit of Fortran, write "DO N TIMES" as "DO N *".

van Wingen: Algol always had free form. Editors can be used to get readability.

Dahlstrand: Fortran 77 is more readable than many other languages. Just add end-of-line comments; other features are too exotic.

Snoek: Free form -- How do you advise compiler implementors to provide a readable listing?

Snoek: Bit data type -- Would like to have it in core.

Dahlstrand: Also favor bit data type in core.

Matheny: But will novice users mis-use it at great cost?

Metcalf: Experienced users are now needing it at great cost.

Schonfelder: Specialists need it very much; "average" users perhaps less.

Dahlstrand: Our users are not "fancy" specialists but need it for foreign data tapes.

Ampt: Short logical is all many users need.

Metcalf: are bit strings to be variable-length?

Greenfield: We should decide whether bit data type is in core, by applying the core criteria. If this won't work, we must change the criteria.

Schonfelder: Variable character length is essential.

Schmitt: Named constants with global scope are needed; e.g., size of arrays.

Also "Stream" input-output; too many record marks in present output.

In favor of variables in format; important though specialized.

Schonfelder: The E format default scale factor should be 1 instead of 0.

A: A switch?

Greenfield: "Undefined" should have a more specific meaning.

Shen: Pointers are needed.

Schmitt: G format goes to E too easily. A better way is needed for specifying this.

7. Large Core vs Small Core, based on question by Jeanne Adams.

See Document 79.

Mangold: Depends on content of Obsolete Features Module.

Bickle: Would like to study Groups and Internal Procedures before deciding whether to eliminate Common and Equivalence.

Ampt: Things are inter-dependent.

Meissner: Everyone says, I want a small core, provided that it contains my favorite goodies.

8. Language Requirements for Numerical Software, by Reid.

See Document 69.

Default Arguments

Dahlstrand: Default input arguments are ok. But how would default output arguments work?

A: Omitted arguments are implemented as local data. Inputs also potentially initialized at each call. Outputs work ok. If results are wanted, just do not omit the actual argument.

Greenfield: Don't like the "adjacent commas" syntax for omitted arguments. Provide only keyword and positional syntax.

Schonfelder: Yes -- "George" has all 3; the adjacent comas are troublesome.

Ampt: You talk about "default" with several meanings -- this is confusing. Default input is quite a different matter from default output. Should dummy arguments have attributes "IN, OUT, or BOTH"?

Shen: This form of explicit control. works well in some languages.

Dahlstrand: Dynamic temporaries in a subprogram use a stack mechanism. This could work just as well for Main program.

Meissner: With regard to problems such as the Graphics package with name conflicts: Can the user get internal names from the CALL as a dummy argument, or from the Package manual?

Schmitt: Default arguments are needed on. a call from "main" program to procedure deep inside the library.

Reverse communication

Meissner: Can guarantee integrity by hiding storage in the library program.

A: Would require dynamic temporary arrays, and has storage inefficiencies.

Schonfelder: All present techniques have problems. "Reverse communication" is inherently un-natural; especially via the EXTERNAL statement.

A: Call with an internal procedure name as an actual argument.

Holberton: Does not solve the data integrity problem, except by copying into dynamic temporaries.

Ampt: You want an internal procedure that is external, but you don't want to declare it EXTERNAL.

Bickle: Previous proposals used explicit "Import" and "Export" to control scoping of names. We should reconsider this as an aid to the needed data protection.

9. Generalized Real Data Type, by Schonfelder.

See Documents 80 and 81.

Schmitt: Use intrinsic functions to inquire about attributes. This could be used to force one object to have a different (e.g., higher) precision.

Dahlstrand: Is edit descriptor related to precision?

A: Not contemplated.

Mangold: Should the letter that replaces E (to designate the precision attribute) appear in the output data?

A: This is a separate question. The proposal so far deals with constants.

Wilson: Type coercion ("widening") could include arguments of generic intrinsic functions.

Reid: Too much widening can be costly in array processing applications.

Holberton: Have a user option for "global" vs "dyadic" widening (i.e., dominant mode for an entire expression vs for the two operands of a single operator).

Ampt: This is related to "optimization level", another user option that is needed. Also we need more study of the Fortran 77 concept of "mathematically equivalent".

Ampt: Generic local variables and arrays?

A: Yes. Declared to have attributes based on an attribute list that appears as a dummy argument.

Adams: Efficiency of compilers?

A: Will have to be solved relative to loaders. Compiler can decide how many versions to compile and how they are to be linked.

Mangold: Some things cannot be resolved until execution time.

A: The simplest answer is to abandon independent compilation.

Reid: When compiling a library, you need to know which versions to generate, based on a data description table. This table might have been generated during compilation of the rest of the code to be used with it. Only the table, not the entire source program, is needed while the library is being compiled.

Ampt: Some extra information could be passed, to defer some resolution until run time.

Dahlstrand: Pass type along with name. This does not conflict with independent compilation. Compare character length being passed by many Fortran 77 systems.

Dahlstrand: "Exact" (vs minimum) precision specification?

A: Considered impractical.

Meissner: Would "simulated decimal rounding" just before each assignment produce the effect desired by numerical analysts? This would mean adding a number, distributed uniformly between -1 and 1, times the specified negative power of 10.

Shen: Inquiry functions for actual attributes? Reconcile with need for processor-independent answers.

The simulated rounding seems to be worth considering.

A: Total portability and reproducibility are not achievable. This proposal looks at "precision attribute" based on a single arithmetic operation. This is not enough to guarantee a specified accuracy for the final result.

Wagener: What other information would be useful if it could be "sensed" during compilation of the calling program?

A: "Precision limits" springs to mind.

Wagener: Have your proposals been inhibited by the assumption that independent compilation is required?

A: Only in restricting the number of variants. Having other information would certainly be useful to the compiler.

10. Interface Mechanisms, by Wagener.

See Document 65.

Extended Call Mechanism

Shen: Can last argument be omitted if default value is wanted? Does this proposal apply to Functions as well as Subroutines?

A: Yes to both questions.

Clarke: Argument checking at compile time?

A: This is included later in the proposal.

Snoek: The user should be able to declare, in the called program unit, that he intends to always omit certain arguments; this would shorten the argument list.

A: Until we heard Reid's presentation, we had not given much consideration to specifying defaults in the called program unit for omitted arguments.

Shen: Algol 60 has a similar facility, except for the default feature (which we like).

Reid: Analogy: COMMON repeats the declaration; GLOBAL points to a single declaration. We could think of calls as having pointers to a single declaration for the arguments, so that attribute matching would happen automatically.

Meissner: The "extended call" proposal started out with a much more limited objective, constrained by not wanting to require recompilation of the called program. We had not worked on this proposal for over a year, until we brought it up again quite recently. We are now looking at the effect of removing the former constraint.

Greenfield: Prefer "block" form of procedure declaration. If declarations appear on both sides of a call, what should happen in case of a conflict? Should the actual argument take precedence? Another possibility is that the declarations could be external to both the called program unit and the calling program unit.

Ampt: We must be very careful in describing "default" semantics.

Sobiesiak: Are these default specifications among the declarations? Can they be selected by macros?

A: Probably.

Kneis: If declarations are external to both program units, additional specification is needed to apply the attributes to the specific program units.

Snoek: The question is related to the idea of "input only" or "output only" dummy arguments.

A: Data base requirements suggest even more kinds of restrictions, that must be specifiable.

Greenfield: Consider permitting the programmer to specify "undefined" as a default.

Schmitt: Some users seem to want actual argument lists with all the facility of input-output lists.

A: Data base seems to need this. Otherwise, we have considerable reservations.

Macros

Shen: Do I need labels? How to resolve multiple calls?

A: This would be a problem with macro-generated DO loops for Fortran 77, where labels are required. The new, loop construct for Fortran 8x, adopted by X3J3, avoids this problem. However, we need to be able to generate other things, such as variables. And there may be other cases where we would still want to generate labels.

Bickle: A macro facility is essential, but we are worried about controlling the ways users will apply it.

A: We, too, have been concerned. Note that the same argument applies to libraries.

Bickle: But we have experience with libraries.

Dahlstrand: Can an argument appear more than once in the expansion? With different results?

Kneis: Macros do not inhibit portability.

Bickle: Macros are a good mechanism for implementing Application Area Support Modules. But it would be dangerous to permit user-defined modules.

Shen: Macro facility should be relatively unsophisticated.

A: Current proposals would permit a macro to redefine a Fortran keyword, but would not permit an Application Area Support Module to redefine Fortran keywords. The X3J3 Working Group (WG 10) now tends to favor a more extensive macro facility.

Using

Rotthauser: Can a core with all the necessary features still be "small, simple, and modern"?

A: We believe this is possible, due to modular architecture.

Shen: Application Area Support Modules must face the problem of Reserved Names.

Sobiesiak: What is the scope of "Using"? It might be useful to permit nesting. Permit a global or more remote reference to an "environment declaration".

A: Some of the facility for this is included in the proposal. Currently the scope of "Using" is considered to be a single program unit. Name management is obviously needed as well.

Kneis: Real time should not be thought of as a separable, self-contained module. It will always be connected with some other application, such as "Real time graphics" or "Real time data base". There is no such thing as pure "real time" Fortran.

11. Conformity Module, by Clarke.

See Document 68.

van Wingen: How much of the requirements should be spelled out in the Standard?

Greenfield: Can this be handled as a separate, collateral Standard?

Meissner: How to handle extensions?

A: Need parallel lists for each extended module.

Dahlstrand: The need for this is increasingly recognized. Would like to consider run-time checking as part of "Exception Handling". The compile-time checks could be done independently, using a separate checking program like PFORT. Would these two mechanisms cover everything that is needed?

A: See page 10. There are several "types of errors", of which non-standard syntax is only one.

Wilson: With regard to declarations for large arrays that might exceed available space, a good approach might be to let the user declare the amount of "Space Available". The Conformity mechanism would then warn the user if that amount of space were exceeded by the requirements of the program.

Holberton: Besides describing all the errors or exceptions that can occur, it would be necessary to have a uniform description of what is to be done in each case. We do not have an adequate model for addressing this question.

A: The proposal addresses the requirement to "report" exceptions. See also page 9 for possible further responses.

Matheny: Concerned with cost. The title "conformity" is better than the former title, "completeness". Standard messages may force a presumed implementation. Experience shows that when a Standard specifies lower limits on complexity of syntax (for example), they are immediately used as upper limits by implementors.

A: These are addressed to some extent under "quantifiable limitations" on pages 10 - 11. These could be negotiated between implementors and users.

Matheny: Limits available in practice will turn out to be larger than anything you could get an implementor to agree to in an advance commitment.

Schmitt: It would be better to permit the user to inquire about the processor limits. Then he could use "space available" to decide how large to declare his array, for example.

A: This is a good idea, but often there are several interdependent processor limits. A workable approach, however, might be to ask for "resource allocations" at the beginning of the source program, to get a warning before the compilation is too far along that it will ultimately fail.

Adams: In what sense is this conformability feature a "module"? It is oriented to the processor rather than the user.

Shen: The "simple inexpensive" criterion is hard to decide. A shorter list would be useful.

For safety, have "assert" statements; "assertion blocks" -- e.g., the range of values taken on by a variable.

Wilson: Would availability of the required Application Area Support Modules be checked?

Meissner: Relation to "exception handling" in the run-tine areas.

A: In some cases, such as with "incremental compilers", there may be no clear distinction between compile tine and run time. Proposal does not preclude integration with an exception handling facility.

Greenfield: How can everyone's efforts be best directed? The Standard should "describe a language". Conformance cannot "guarantee portability". We could do this as a "collateral standard" with requirements that are stricter than those of the main language, in the direction of being more nearly able to guarantee portability. The precise mechanisms would not have to be spelled out in such a standard.

Schmitt: If messages were standardized, we should permit an analogous set of messages in other languages. This might be done once for each language. This would argue in favor of standardizing the messages.

Dahlstrand: Fortran 77 permits extensions in certain areas. This could be made more restrictive upon request by the user.

A: That would imply that each implementation must include such a tool. This may be too much to ask.

Muxworthy: How to handle "conformability" as a collateral standard?

Greenfield: The separate document could be more restrictive.

Dahlstrand: But conformity should have the tightest restrictions.

Mangold: Relation to portability? This proposal, e.g., maximum integer, might be too large for some machines. Standard error messages mean standard parsing, or else would still leave a lot to be processor-dependent.

Straw vote: "The proposed Conformity Module should be developed further". 12 yes; 1 no; 11 undecided.

12. Real Time Fortran, by Kneis.

See Document 66.

Holberton: Some additional information should be provided to SC-5 members, along with EWICS request for vote to adopt IRTF as new work item.

Meissner: What does "to mask" or "to unmask" an event-mark mean?

A: Think of "to hide" or "to reveal" the event-mark, with respect to the program. 'The event-mark still gets set, but the program is not aware of is while the event-mark is in the "masked" state.

13. Implementation of Core and Modules, by Snoek.

See Document 83.

Wagener: Recent discussions in X3J3 Working Group 10 seem mostly consistent with this paper. A small point is whether there could be more than one "standard" Language Extension Module to provide more flexibility in implementing some parts of the language without others.

A: That does not seem objectionable.

Wagener: Extensions that are unique to an Application Area Support Module: Are these really part of the Application? In our terminology, any Language Extension Module would be available to all Applications.

A: If an extended feature is of specialized interest, it should be attached more closely to the Application than to the Language Extension Module. Putting it inside the Application Area Support Module will hybridize the structure.

Ampt: This may be a matter of terminology. The extensions may need X3J3 approval even when they are only for a specialized area.

Kneis: There should not be a need for Language Extension Modules specific to certain Applications. Extensions should be generalized so other Applications can also use them.

A: This is true for the most part. However, there may be some exceptions.

Bickle: Agree with Kneis. The resolution of these questions will depend on actual examples, not on abstractions.

If you insist that Applications be "independent", how can this be monitored?

Sobiesiak: The proposer must divide extensions from the rest of the module. Proposers with common problems should be asked to resolve differences among themselves.

Shen: Real Time Fortran is not just pertinent to a small area of application. Compare IRTF with Pearl and other real-time languages. This reveals some areas that must definitely be considered in future work.

The 1980 proposal on "scheduling" does not address "priority".

It is not clear whether the same process can have multiple incarnations.

What corresponds to the "system division" of Pearl? (ADA also neglects this area.).

Distributed systems are not addressed. (This is related to the previous point.).

Synchronization is a generally useful extension. This should not be relegated to a "corner" of the language. ["Message exchange" synchronization, such as COBOL uses, operates at a "higher level" than "Semaphore" synchronization. This could make a good "general extension" feature; e.g., needed for "transaction" synchronization in database applications.].

Kneis: Only "Process input and output" is special. All other parts of IRTF are more general. "Priority" is part of our scheduling, but we schedule a "virtual" processor, while the "actual" processor is the place where most synchronization problems arise. The same applies to multiple incarnations. "System" considerations mainly relate to the "actual" system, with which we are less concerned.

Greenfield: Names may be more important than many people realize. "Synchronization of independent processes" is the essence; this is useful in a very general context.

Wagener: We have to carefully review the placement of various language elements. Otherwise, "accidental" timing of proposed extensions and applications will have too great an influence on the eventual shape of the result.

We have thought of Application Area Support Modules as a reasonable way to accommodate "external" groups. X3J3 must continue to review these to identify features that have wider applicability. Snoek's suggestion (to separate special features between "special Language Extension Modules" and Application Area Support Modules) is good, in that the proposer can thus help to call such generally useful features to the attention of X3J3.

Adams: All proposals that involve "new syntax" must be brought to x3J3.

Greenfield: Application Area Support Modules will not necessarily be standardized in synchronization with basic Fortran revisions. Some parts of these might be added to Fortran anyway at the next revision. This implies close collaboration in the development of such parts.

14. A User's View of Real Time Fortran, by Dannijs.

See Document 84.

Greenfield: Extension to distributed systems will be the trend, rather than having a single process "in control" at any time.

Meissner: Many ideas from real-tine suggest a framework for more general Fortran problems. For example, asynchronous input and output, or exception handling.

Kneis: We have not explicitly provided a forced immediate "task abort". Rather, a way to instruct a task to terminate itself "gracefully".

15. General or Specific Multi-Tasking Semantics? by Ampt.

See Document 88.

Kneis: Agrees that EWICS-Fortran must move toward much more generality and language independence. The models now in hand are more specific, for reasons that are mainly historical.

Holberton: In a procedure reference, each separate argument is in effect a separate "function". This suggests a possible framework for sequencing the effects of various arguments. However, this concept is not too compatible with language independence.

Wagner: Agrees that Application Area Support Module designers should first broadly and abstractly describe the order and discipline of an Application area, without at first using specific language-dependent syntax. Then later, suggest Fortran-related implementation details. Thus, they would be using a "top-down" design approach.

Matheny: X3J3 has been having some effect on "defining" these processes. There is another way to look at multi-tasking, as management of several independent tasks on a single piece of hardware.

Signor: X3J3 must maintain a responsibility for awareness of common needs, and for performing synthesis where needed.

16. Role of the US National Bureau of Standards, by Holberton.

Summary: The National Bureau of Standards is the agency, within the United States Government, that has primary responsibility for programming language standards support.

Fortran 77 was adopted as a US Federal Information Processing Standard, on 4 September 1980. This means that federal procurements after March 1982 that involve Fortran must be based on Fortran 77, unless a special waiver is. obtained.

The National Bureau of Standards is supporting several projects by contract to other agencies. one of these is a Fortran 77 analyzer, which will eventually be available from the National Technical Information Service. A specification has been written for the development of a suite of compiler test programs, but only one project has been contracted for to date, involving one-third of the work. Another contract supports a portability study at Salford University. Related projects include study of ways to spread the use of software tools in industry.

Fortran 77 analyzer

Dahlstrand: When the Fortran 77 analyzer "reports deviations", will this include extensions, limitations, specifications ("integer lengths"), etc?

A: Only what is covered in the Standard.

Bickle: Are results published?

A: Yes -- a list of compilers tested, with indication of whether or not each passed. There is COBOL precedent for this; in fact, COBOL now provides an annual rectification.

Test Suite

Shen: How can you ensure completeness?

A: We have taken some steps to this end.

Conversion

Dahlstrand: Can conversion costs be taken into account in evaluating the proposals of hardware suppliers?

A: No. The aim is to encourage users to write programs that are easily convertible.

Other references

"Validation, Verification and Testing for the Individual Programmer" (NBS Special Publication 500-56) available from Government Printing Office, Washington DC 20402. Ordering number is 003-003-157-8; price is $1.75 per copy.

17. Generalization, by Dahlstrand. See Document 76.

"The most general statement that can be made is silence."

Undeclared variables As a difficulty in a general "name hierarchy" (a la Simula); user should be able to invoke a "safety module".

Dynamic array bounds

Shen: Restricting this to subprograms will enable stack management.

Meissner: This is only apparent. Stack management releases storage only at procedure return; this never happens for the main program (until end of execution). Otherwise, there is no real difference.

Snoek: PL/1 permits dynamic arrays in the main program also.

Exception handling

Matheny: Should "resume" be provided as a possible option?

A: It is needed.

General

Shen: Against variable number of arguments -- hard for compiler to check.

A: We may not need this feature so much, when we have data structures.

Shen: We need both "Until" and "While".

A: Use of Algol 60 has turned up some difficulties with these. Current X3J3 position is indefinite looping with exit.

18. Discussion of the August 1978 Guidelines.

See Page 2 of Document 67a.

Munchhausen: DIMENSION is no longer needed, and should be deleted.

Shen: What replaces Alternate Return?

Greenfield: Return a value.

Kneis: Alternate Return leads to unsafe programing. Should be discontinued eventually. This is what "Obsolete Features Module" mans: no longer available after 1999.

Matheny: Exception handling, based on subprogram call, is needed as a more specific feature to replace most of the need for Alternate Return.

Mangold: How are you assuring upward compatibility when a feature is replaced? For example, old and new source form.

Adams: We are considering source form as a special case.

Holberton: Next Standard will have to more carefully describe the form of a "line".

Bickle: The proposed additions and deletions are ok, if adequate replacements are provided. Worried especially about Common and Equivalence.

Group 1

Additions: Free Form Source; Larger Character Set; Longer Names.

Deletions: Column 6 Continuation; C for Comment.

Ampt: Each specific item has some concerns. Doubt that longer *names will be a burden on the compiler.

Snoek: It is bad to mix free-form with a "line" concept. If end-of-line is special, beginning could also be special.

Mangold: A specific set of record lengths, that can be processed, is needed.

Muxworthy: We are already using a free form continuation syntax very much like this one, and we like it.

Dahlstrand: Short names are a real problem. I like mnemonic names. The other features are not urgent.

Group 2

Additions: Data Structures; Array Processing; Global Data Definition.

Deletions: Equivalence; Common and Block Data; Passing an Array Element to a Dummy Array.

Ampt: Array operations should be kept very simple.

Dahlstrand: Passing an array element to an dummy array causes no real problems. It is a nice feature for bypassing rules. We can't predict how people will use features.

Equivalence has many purposes; therefore its replacement may require at least 3 different features. Agree with keeping arrays simple.

Shen: What are the details of data structures? Can a "form" be passed as an argument? Also, we need pointers. We need other array intrinsics, especially a "swap".

Matheny: We plan to permit passing a "form". Pointers can be dangerous.

Muxworthy: Alias is needed, as part of the replacement of Equivalence.

Vossenstijn: We need array-valued functions.

Greenfield: Some uses of pointers can be achieved with subscripts to a vector of structures.

Mangold: Fortran could become an array language. Interfaces can be developed to convert many DO loops.

Some of these features are parts of "historical Fortran".

Schmitt: If you don't support Heap storage management, you don't need pointers. The features added so far do not require Heaps. If pointers are added, they should be severely restricted.

With regard to Global Data, this feature should include all attributes, including space allocation.

Group 3

Additions: Control Structures (Looping and Case); Internal Procedures.

Deletions: Arithmetic IF; Computed GO TO; Alternate Return; Assign and Assigned GO TO; Statement Functions.

Schmitt: Don't invent too many kinds of loops. Causes confusion and compiler redundancy.

Shen: Control part of a loop could include an vector of integer Values, to be used as a list of values to be assumed by control variable.

Possible problems with SAVE and DATA statements if recursion, is permitted. Need "Initialize" statement.

Ampt: In favor of small core. Can macro facility be kept simple?

Wagener: Expect macro not to be in core.

Snoek: Internal procedures can be error-prone.

Mangold: Be more careful what you add. You may have to take it out next time. For example, Alternate Return went in with Fortran 77, is now already Obsolete.

Meissner: Stability of core is a serious problem.

Bickle: Multiple entry and return are being implemented by everybody. All should do it the same way.

Matheny: We want to delete any required storage association, such as is implied by Fortran 77 Common.

Group 4

Additions: Precision Specification.

Deletions: Double precision.

Ampt: More is needed, but not too extensive. Current proposal seems hard for the general user to understand.

Rotthauser: Don't need precision in core. No better for average user than what is now in Fortran 77. Would not enhance portability. But it would be nice as an extension.

Kneis: Agree that this is not needed in core. Real time needs to extend this.

Dahlstrand: Precision may have some problems, but if done should be a core feature. Provide simple defaults for the naive user.

Ampt: "Automatic" (generic) type conversion (coercion) is dangerous.

Group 5

Deletions: ERR=, END =; Edit Descriptors H, X, D; Specific Names for Intrinsic Functions; Entry Association.

Schmitt: If deleted, need to standardize Iostat values. It already hampers portability in Fortran 77.

Mangold: This whole feature should be replaced by a proper exception handling capability. There are synchronization problems with having Iostat in a separate statement, that do not happen with error specifiers in the same statement; since the system can guarantee within-statement synchronization integrity.

Holberton: Alternate Return is analogous to END=.

Matheny: Iostat provides the same information as END= or ERR=.

Ampt: A good solution, in the multi-tasking environment, is an exception handling facility.

Dahlstrand: This area needs more "brainstorming". we should not inhibit any ideas at this stage.

General

Dahlstrand: This entire proposal is a combination of many "nice" features. Many difficult choices are required. Good luck.