Minutes of the interim ISO/IEC JTC1/SC21/WG7/E-LOTOS meeting Liège, 18th-21st of December 1995

0. Attendance list

Arnaud Février, E.N.S.T., France, fevrier@email.enst.fr
Hubert Garavel, INRIA, France, Hubert.Garavel@imag.fr
Christian Hernalsteen, Free University of Brussels, Belgium, chernals@ulb.ac.be
Alan Jeffrey, University of Sussex, United Kingdom, alanje@cogs.susx.ac.uk
Guy Leduc, University of Liège, Belgium, leduc@montefiore.ulg.ac.be
Luc Léonard, University of Liège, Belgium, leonard@montefiore.ulg.ac.be
Elie Najm, E.N.S.T., France, najm@res.enst.fr
Krimo Nimour, E.N.S.T., France, krimo@email.enst.fr
Charles Pecheur, University of Liège, Belgium, pecheur@montefiore.ulg.ac.be
Juan Quemada, Tech. Univ. of Madrid, Spain, jquemada@dit.upm.es
Mihaela Sighireanu, INRIA, France, Mihaela.Sighireanu@imag.fr
Bernard Stepien, University of Ottawa, Canada, bernard@csi.uottawa.ca

1. Appointment of a secretary

Elie Najm volunteered for Monday to Wednesday, and Luc Léonard for Thursday.

2. Agenda

The following agenda is agreed on:

- List of input documents
- Rapporteur's report
- Presentation of input documents
- Technical discussions (data model, modules, behavioural model)
- Planning
- Liaison and co-ordination
- Closing

3. List of input documents

LG1:	Defect report concerning IS8807(LOTOS) and proposal for a correct flattening of LOTOS parametrized types (AFNOR)
LG2:	Application of the proposed E-LOTOS datatype language to the description of OSI and ODP standards (AFNOR)
LG3:	French-Romanian comments regarding some proposed features for E-LOTOS data types (French/Romanian experts)
LG4:	A Proposal for the datatype part of E-LOTOS applicable to the formal description of OSI and ODP standards (French/Romanian experts)
LG5:	A wish list for the behaviour part of LOTOS (French expert)
LG6:	A short comment on LG2 (UK expert)
LG7:	Semantics for a fragment of LOTOS with functional data and abstract datatypes (UK expert)
LG8:	Compositional specification of ODP binding objects (Belgian/French experts)
LG9:	A suspend-resume operator for TE-LOTOS (Belgian/French experts)
LG10:	Defining equivalence between time/action graphs and timed action graphs (Spanish experts)
LG11:	Specifying the suspend-resume behaviour with time in LOTOS (Canadian Expert)
LG12:	The suspend Resume operator application to action interrupts - Canadian Experts

4. Rapporteur's report

Work on datatypes has been going on by many experts in the interim period. The PDAD was delayed for one year in Ottawa. Thus, a quick convergence to a standard is a condition of the success of E-LOTOS. We should also start taking decisions, especially regarding datatypes, and provide the kernel language.

5. Presentation of input documents

Hubert Garavel presents LG1: this document is a proposal for a technical corrigendum of the LOTOS standard. It corrects a mistake in the IS8807 document, in the definition of the flattening process and more specifically in the cases where renaming is used in conjunction with actualisation. Decision was taken that this mistake should be avoided in E-LOTOS. It was decided that Hubert Garavel works on issuing a formal defect report using the official procedure.

Hubert Garavel presents LG2: this document deals with the application of the E-LOTOS data types as defined in LG4. The examples are LOTOS specifications taken from OSI. These are: CCR, TP, two ODP applications (metamovie and video on demand), LAPB protocol. The document of Annex A presents a set of possible design choices. The specifications of LG2 use the following choices: no polymorphism, no anonymous tuples, overloading is allowed, modules à la Lotosphere. The conclusion of this work is that the modified specifications are much shorter. Lengthy pieces of specifications where shortened due to the use of projections and updating functions on unions of records. Moreover, defects were found in the specifications and reported to ISO.

Alan Jeffrey presents LG6: this document argues that projection is not needed on union records if structuring is used appropriately. There are problems in refining data types in union records.

Hubert Garavel presents LG3 which compares the two solutions of Annex A and gives arguments in favour of proposal LG4.

Hubert Garavel presents LG4: a proposal for data types which attempts compatibility with a subset of IDL, by providing in/out parameters and exceptions for functions. However, interface references are not taken into account. LG4 presents also the abstract syntax and static semantics of data type language. It separates primitive constructs from shorthand notations. Dynamic semantics is defined denotationally.

The experts recognized that it is important to have good mappings between E-LOTOS and other languages standardized within ISO SC21 and SC22 including IDL (either directly or by syntactic sugar).

Alan Jeffrey presents LG7: the aim of this document is to provide a unified approach for the semantics of the data part and the behaviour part. It provides a uniform approach for defining the semantics using environments and logic rules for both data and behaviour. Subtyping is one of the features aimed at by the proposal, although it is not yet covered by it. The document also discusses the issue of the communication of abstract data types whereby the implementation should be hidden. It hints at three solutions to this issue:

- export the implementation
- ban the communication of abstract types
- allow a user specified equality and reachability.

The proposal includes a small discussion on a possible extension of gates as first class citizens and how this extension encodes the Pi-calculus.

Hubert Garavel presents LG5: the document provides a list of 19 enhancements to the process part of LOTOS. The main improvements aim at unifying the syntax between data and process parts of LOTOS (for instance by introducing exceptions). The improvements will be discussed later during the technical presentations.

Guy Leduc presents LG8: a preliminary version of this work had already been presented in Ottawa. It describes and ODP binding object. Two descriptions were provided: in MT-LOTOS and ET-LOTOS. The document does not contain a full comparison of the two specifications. A quick assessment shows that the ET-LOTOS specification implies some extra technicality but allows the full use of the multiway rendezvous. On the other hand, MT-LOTOS provides for more reusable modules and allows a meaningful typing of gates.

Arnaud Février presents LG9: the document presents a Suspend/Resume extension. This enhancement is shown to be meaningful in a time context. It features two different ways of letting time pass in processes. Suspend/Resume is interesting in conjunction with the M-LOTOS extension.

Bernard Stepien presents LG12: the Suspend/Resume proposed in this document was applied to the flight warning computer example. Time is shown to be important for this extension.

Bernard Stepien presents LG11: the Suspend/Resume is made compatible with time. Aging in the suspended process is solved by the introduction of a new Hold operator. Hold appears only in the semantical level and need not to be seen by the user. The present proposal should be compared with the one of LG9.

Juan Quemada presents LG10: it is a new proof of the equivalence of the two semantics provided for timed LOTOS (Spanish and Belgian). It will be examined by Belgium.

6. Datatypes

Documents LG4 and LG7 where the basis of a discussion on datatypes. Hubert Garavel and Mihaela Sighireanu present proposal LG4. The following simple example was considered:

```
Function SUB(x,y:real) is
x-y
endfunc
```

and the different notations for argument (positional, named fields) in function call was discussed. LG4 allows both positional and field naming notations for arguments in the function call, e.g.:

```
SUB(0,1)
SUB(X:=1, Y:=0)
SUB(Y:=0, X:=1)
```

Moreover, LG4 provides many convenient features in an integrated way, including:

- ellipsis notation “...” in patterns,
- ellipsis notation “...” in function/constructor calls,
- easy interfacing with standardized computer languages,
- automatically defined projections and update functions on both records and unions

A comparison was made with LG7 where functions are defined either with positional calling or with field name calling. Record subtyping concerns are the reason for the two different function declarations of LG7. Record subtyping is discussed in the Ottawa output document but not in LG7.

A discussion led to the conclusion that ADA function call styles can be integrated with LG7.

A discussion followed about interfacing specifications to the external world and on how IDL specifications can be used in conjunction with anonymous records.

Record subtyping (implicitly considered in LG7) is seen to be an issue and a difference with LG4. Record subtyping allows an easy definition of functions on projections of field records. However, the projection and updating operators defined in LG4 reduce the need for such functions. Record subtyping could be extended to allow gate subtyping, which is a mechanism for providing modular process composition. Projections on unions of records present problems in conjunction with record subtyping. Another difference between LG4 and LG7 resides in the feature of anonymous tuples (proposed in LG7). Anonymous tuples allow not having to name types used once, and thus to have concise composition of functions returning more than one result.

The sync construct of LG7 was discussed. This operation is very powerful but its impact on the behaviour language has to be evaluated. It was noticed that if this construct is used then very basic properties, like commutativity and associativity of the parallel operators, are dependent on well-behaved user specified code. The sync operator will be replaced by the use of quotient types (similar to Act One equations) in the output document of this meeting.

A discussion on LG7 led to the conclusion that modules and record extensions may provide a replacement solution for record subtyping. In this case, the order of fields in record definition of LG7 may be also reconsidered: records are order sensitive in all cases except in function application and some other cases.

An agreement was reached which is to use LG4 as the basis for the user oriented language and LG7 for the core language. The semantics of LG4 will be defined through an LG4 $\rightarrow$ LG7 translation. In that case, LG7 should be extended to include record extensions and exceptions. The precise details of this agreement will be worked out before the next meeting.

7. Modules

The group discussed the list of questions provided in Annex A of the Ottawa output document. Answers were provided as follows:

Q1.1 - Need for separate interfaces: Yes
Q1.2 - Several modules for a given interface: Yes and possibly in different languages.
Q1.3 - Several interfaces for a given module: Yes (provided that signatures are well defined)
Q1.4 - Synthesising the interface from the module: For further study.
Q2.1a - Objects declared in a module: Anything that can occur in a global level declaration (with the possible exception of modules and interfaces)
Q2.1b - Objects declared in an interface: Same as Q2.1.a
Q2.2a - Declaration of algebraic equations in modules: Yes
Q2.2b - Declaration of algebraic equations in interfaces: Yes
Q2.3a - Can modules contain definitions of modules (and interfaces): For further study.
Q2.3b - Can interfaces contain definitions of interfaces (and modules): For further study.
Q2.4 - How are types declared in interfaces: Export of type names only or complete data type declarations.
Q2.5 - Profile information of constructors to appear in an interface: Full data type declaration

Q2.6 - Profile information of functions to appear in an interface: Full header but not body.
Q2.7 - Profile information of processes to appear in an interface: Full header but not body.
Q2.8 - Where are modules and interfaces declared: Only at global level.
Q3.1a - Can modules be declared in a cyclic manner (not only in DAGs): For further study.
Q3.1b - Can interfaces be declared in a cyclic manner (not only in DAGs): For further study.
Q3.2a - Allow algebraic module expressions: For further study.
Q3.2b - Allow algebraic interface expressions: For further study.
Q3.3a - Allow arbitrary module expressions: For further study.
Q3.3b - Allow arbitrary interface expressions: For further study.
Q4.1a - Locality of objects in a module to be declared implicitly or explicitly: implicit declaration is agreed. No agreement on explicit declaration.
Q4.1b - Objects local to an interface, does it make sense ? No.
Q4.2a - Is hiding a possible combinator in module expressions: For further study.
Q4.2b- Is hiding a possible combinator in interface expressions: For further study.
Q4.3 - Redefinition of exported objects: For further study.
Q4.4 - Export of constructors of a type: Either export all or none.
Q4.5 - Export of the built-in syntactic equality on functions: For further study.
Q5.1a - Import of modules in other modules: For further study.
Q5.1b - Import of modules in other interfaces: For further study.
Q5.2a - Import dependency graph (module): For further study.
Q5.2b - Import dependency graph (interface): For further study.
Q5.3a - Importing a subset of a module in a module: OK for positive selection. Negative selection for further study.
Q5.3b - Importing a subset of an interface in another interface: OK for positive selection. Negative selection for further study.
Q5.4a - Possibility of enrichment of a module with another module and thus have the need to ensure consistency: The new proposals for data types have a built-in consistency.
Q5.4b - Possibility of enrichment of a interface with another interface and thus have the need to ensure consistency: The new proposals for data types have a built-in consistency.
Q5.5a - Should modules be closed : For further study.
Q5.5b - Should interfaces be closed : For further study.
Q5.6a - Rules for solving clashes in case of multiple module importation: For further study.
Q5.6b - Rules for solving clashes in case of multiple interface importation: For further study.
Q6.1a - Renaming functionality for modules: Yes (exact mechanism for further study).
Q6.1a - Renaming functionality for interfaces: Yes (exact mechanism for further study).
Q7.1 - Syntax for object names in module: For further study.
Q8.1 - What kind of objects can be used as module parameters: For further study.
Q8.2 - Generic interfaces: No, the typing for parametric modules is given by the signature of their parameters and the signature of their result.
Q8.3 - Means to specify formal objects: Through interfaces (à la Lotosphere or SML).
Q8.4 - Partial instantiation of actualization: For further study.
Q9.1 - How to import external objects: Interface should not be external. Modules can be in E-LOTOS or other languages. No mixing of internal and external objects in a module. Generic external modules are for further study.
Q9.2 - Support of separate compilation in E-LOTOS: It is a desirable feature but technical problems are left for further study.
Q10.1 - Maintenance of compatibility with Act-One: Yes, as much as possible.

Q10.2 - Compatibility with LOTOS "specification" declaration: A specification is a module with a distinguished process. Also, the possibility for data specification with a distinguished function is left for further study.
Q10.3 - Should modules be the only means for having nested declarations of processes and data types: Yes.
Q11.1 - Semantics of modules handled completely in the static semantics phase: Analysis of specifications is done in three aspects: type checking, module flattening and execution semantics. The first two aspects can be dealt with at compile time.
Q11.2 - Keep the flattening approach for modules: Yes, with the possible exception of module identifiers.
Q11.3 - Style of the static semantics of modules: Both declarative and algorithmic (reference implementation) styles can be used.

Alan Jeffrey introduced three extra questions:

Q.AJ.1 - Forbid overloading within a module: For further study.
Q.AJ.2 - Allow for a restricted form of polymorphic instantiation of generic modules (e.g. use of import of generic modules with a kind of “any” parameter): A concrete proposal should be provided and examined.
Q.AJ.3 - Allow for parametrization of generic modules with generic modules: A concrete proposal should be provided and examined.

8. Process part

Bernard Stepien presents the motivations behind the proposal of LG11 and LG12. The need for the suspend/resume operator has been recognised. The operator should be simple and should have a well-defined time semantics. In the first Canadian proposals time was not considered as an issue. LG11 addresses the time passing semantics in suspended processes, and made the design choice of letting time pass.

Arnaud Février and Christian Hernalsteen present LG9 which is an alternative proposal.

The two approaches were compared. An important difference with LG11 is that, in LG9, time is frozen in a suspended process. Based on the examples of real-time schedulers, we finally gave preference to the time freezing approach, but we agree that further investigation and examples can be useful to decide on this issue. A last point was raised by LG9, which is the possibility for a process to suspend itself. This, also, will be further investigated.

Hubert Garavel and Alan Jeffrey hinted at the possibility that suspend/resume operators may be special cases of a parallel operator with priorities. The feasibility of this approach remains for further investigation.

Hubert Garavel gives an overview of some of the 19 proposals contained in LG5.

Proposal 4: Introducing two “case” operators

The group agrees that a general “case” statement with pattern matching and priority is desirable in the language. The problem of the syntax of match expressions for patterns is left for further study. The usual “case” being syntactic sugar on top of the general “case” statement, it is acceptable.

Proposal 16: Removing the “where” clause from process definitions

This proposal has already been agreed in the discussion on modules. Consequently, proposals 18.1 and 19.1 are rejected.

Proposal 5: Introducing an “if-then-else” operator

The “if-then-else” operator being syntactic sugar above the general “case” and being present in most other languages, its principle is accepted. It is agreed that the “else” part should be mandatory to be aligned with the “if-then-else” in the data part.

Proposal 12: Unifying the “;” and “>>” operators syntactically

There is no agreement about syntactical unification between action prefixing and enabling. On one hand, it is desirable to have a single operator to express sequential composition. On the other hand, there are concerns about potential syntactical ambiguities. The point is left for further study.

Proposal 13: Unifying the “;” and “>>” operators semantically

We agreed on the introduction of a new sequential composition operator, that avoids the “i” action in sequential composition. The enabling can be expressed in terms of this new operator and should be kept as a shorthand. If it is possible, we consider that unifying semantically action prefix and the new sequential operator is even more desirable. However, no such proposal exists for the time being. This last point is thus left for further study.

Proposal 6: Extending the “let” operator

The “let” operator being syntactic sugar above the general “case”, its principle is accepted.

Proposal 10: Introducing a “par” operator on finite value domains

The group agrees that the principle of a general “par” operator parametrized on a finite data domain is desirable. The finiteness of the data domain has to be checkable at the static semantics level, and therefore the way to define the data domain is left for further study.

The remaining proposals of LG5 have not been discussed by lack of time.

9. Planning

The ISO/WG7 Kansas City meeting is 13 - 24 May. We aim at producing a new version of the base document by March 15. Therefore, contributions to this document should be sent by the experts to Juan Quemada before this date.

Contents of the Liège output document

Annex A - Data Types:
- Part 1: Reference to the Discussion document (previous annex A)
- Part 2: User level language (edited by France and Romania)
    - Abstract syntax
    - Static semantics (based on attribute grammars)
    - Dynamic semantics (denotational)
- Part 3: Core language (edited by UK)
    - Abstract syntax
    - Static semantics (sequents)
    - Dynamic semantics (big step semantics)
- Part 4: Translation between user and core languages
  (edited by France, Romania and UK)
- Part 5: Relationships between the core language and the behaviour part
  (edited by UK)

Annex B - Tagged Typed gates: kept as is
Annex C - Time: kept as is
Annex D - Mobility: kept as is
Annex E - Modules: it is the list of questions/answers listed in section 7 above.
Annex F - Operators:

- Part 1 is kept as is
- Part 2 is replaced by an updated version of LG5.

Annex G - Integration issues:

- Part 1 is kept as is
- A part 2 will be added: MT-LOTOS - integration of time and mobility
- A part 3 will be added: document LG8.

Besides the Liège output document, contributions on the following issues are expected:

- Concrete proposals on modules
- A concrete proposal for the datatype base environment in the user-level language
- A detailed comparison of LG9, LG11 and annex G (part 1) regarding the suspend/resume operator, and possibly a unified proposal for this operator.
- A detailed comparison of annex F1 (generalized termination and enabling) and the exception mechanism proposed in LG5, and possibly a unified proposal.

10. Liaison and co-ordination issues

It was decided that, in the next meeting in Kansas City, we should have a common meeting with the architectural semantics group.

11. Closing

The meeting was closed on Thursday 21 at 16:30.