

First edition
2001-08-15

**Information technology — Enhancements
to LOTOS (E-LOTOS)**

Technologies de l'information — Mises en valeur de LOTOS (E-LOTOS)

Reference number
ISO/IEC 15437:2001(E)



© ISO/IEC 2001

PDF disclaimer

This PDF file may contain embedded typefaces. In accordance with Adobe's licensing policy, this file may be printed or viewed but shall not be edited unless the typefaces which are embedded are licensed to and installed on the computer performing the editing. In downloading this file, parties accept therein the responsibility of not infringing Adobe's licensing policy. The ISO Central Secretariat accepts no liability in this area.

Adobe is a trademark of Adobe Systems Incorporated.

Details of the software products used to create this PDF file can be found in the General Info relative to the file; the PDF-creation parameters were optimized for printing. Every care has been taken to ensure that the file is suitable for use by ISO member bodies. In the unlikely event that a problem relating to it is found, please inform the Central Secretariat at the address given below.

© ISO/IEC 2001

All rights reserved. Unless otherwise specified, no part of this publication may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm, without permission in writing from either ISO at the address below or ISO's member body in the country of the requester.

ISO copyright office
Case postale 56 • CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 749 09 47
E-mail copyright@iso.ch
Web www.iso.ch

Printed in Switzerland

Contents

1	Scope	1
2	Conformance	1
3	Normative reference	1
4	Terms, definitions and notation	1
5	E-LOTOS grammar	2
5.1	Lexical Structure	2
5.1.1	Character set	2
5.1.2	Comments and separators	3
5.1.3	Identifiers	3
5.1.4	Reserved words	3
5.1.5	Identifiers classes	4
5.1.6	Non-terminals classes	4
5.2	Syntax of the language	5
5.2.1	Specification	5
5.2.2	Top-level declaration	6
5.2.3	Module body	6
5.2.4	Module expression	6
5.2.5	Module formal parameters	6
5.2.6	Interface expressions	6
5.2.7	Record module expression	6
5.2.8	Interface body	7
5.2.9	Formal parameter list	7
5.2.10	Renaming/Instantiation	7
5.2.11	Equations declaration	7
5.2.12	Simple equations declaration	8
5.2.13	Declarations	8
5.2.14	Expressions	8
5.2.15	Expression Atoms	8
5.2.16	Behaviour expressions	9
5.2.17	Disabling behaviour expression	10
5.2.18	Synchronization behaviour expression	10

5.2.19	Concurrency behaviour expression	10
5.2.20	Selection behaviour expression	10
5.2.21	Suspend/Resume behaviour expression	10
5.2.22	Interleaving behaviour expression	10
5.2.23	Behaviour term	10
5.2.24	Behaviour atom	11
5.2.25	Type expressions	12
5.2.26	Record type expressions	12
5.2.27	Value expressions	12
5.2.28	Record value expressions	13
5.2.29	Patterns	13
5.2.30	Gate parameter list	13
5.2.31	Actual parameter list	13
5.2.32	Exception parameter list	13
5.2.33	Record patterns	14
5.2.34	Behaviour pattern matching	14
5.2.35	Expression pattern matching	14
5.2.36	In parameter list	14
5.2.37	In parameter	14
6	E-LOTOS abstract syntax	14
6.1	Overview	14
6.1.1	Syntactic sugar	15
6.1.2	Abstract syntax	15
6.2	Concrete to abstract syntactic translation	16
6.2.1	Interface body	16
6.2.2	Formal parameter list	18
6.2.3	Declarations	18
6.2.4	Expressions	21
6.2.5	Expression atoms	21
6.2.6	Behaviour expressions	24
6.2.7	Interleaving behaviour expression	24
6.2.8	Behaviour atom	25
6.2.9	Type expressions	31
6.2.10	Record type expressions	32
6.2.11	Record value expressions	32
6.2.12	Gate parameter list	33
6.2.13	Actual parameter list	33
6.2.14	Exception parameter list	33
6.2.15	Record patterns	33
6.2.16	In parameter list	34
6.2.17	In parameter	34
7	E-LOTOS semantics	35
7.1	Overview	35
7.2	Static Semantics	36
7.2.1	Static semantic objects for Base	36
7.2.2	Judgements on static semantics for Base	37
7.2.3	Extended identifiers	38
7.2.4	Static semantic objects for Modules	39
7.2.5	Judgements on static semantics for Modules	39
7.2.6	Cycle freedom	40
7.2.7	Context morphism	40
7.2.8	Realization	40

7.2.9	Interface Instantiation	40
7.2.10	Interface Matching	41
7.2.11	Renaming/Instantiation	41
7.3	Untimed dynamic semantics	44
7.3.1	Untimed dynamic semantic objects for Base	44
7.3.2	Judgements on untimed dynamic semantics for Base	45
7.3.3	Dynamic semantic objects for Modules	46
7.3.4	Judgements on untimed dynamic semantics for Modules	47
7.3.5	Environment morphism	48
7.3.6	Signature Instantiation	48
7.3.7	Renaming/Instantiation	48
7.4	Timed dynamic semantics	48
7.4.1	Judgements on timed dynamic semantics	49
7.5	Write-many variables: the value substitution operator	49
8	The E-LOTOS modules	52
8.1	Specification	52
8.1.1	Specification	52
8.2	Top-level declaration	53
8.2.1	Module not constrained by an interface	53
8.2.2	Module constrained by an interface	53
8.2.3	Generic module not constrained by an interface	54
8.2.4	Generic module constrained by an interface	55
8.2.5	Interface declaration	55
8.2.6	Sequential top declaration	56
8.3	Module body	56
8.3.1	Block declaration	56
8.3.2	Module Expression	57
8.4	Module expression	57
8.4.1	Module aliasing not constrained by an interface	57
8.4.2	Module aliasing constrained by an interface	58
8.4.3	Generic module actualization not constrained by an interface	58
8.4.4	Generic module actualization constrained by an interface	59
8.4.5	Generic module renaming/instantiation	59
8.5	Module formal parameters	60
8.5.1	Single	60
8.5.2	Disjoint union	60
8.6	Interface expressions	61
8.6.1	Interface identifier	61
8.6.2	Simple renaming	61
8.6.3	Explicit body	61
8.7	Interface body	62
8.7.1	Type hiding the implementation	62
8.7.2	Type synonym	62
8.7.3	Constructed type	62
8.7.4	Named record type	63
8.7.5	Process declaration	63
8.7.6	Equations	64
8.7.7	Sequential declaration	64
8.8	Record module expression	64
8.8.1	Single	64
8.8.2	Disjoint union	65
8.8.3	Renaming tuple	65

8.9	Equation declarations	65
8.9.1	Equations declaration	65
8.9.2	Sequential	66
8.10	Simple equation declaration	66
8.11	Declarations	66
8.11.1	Type synonym	66
8.11.2	Type declaration	66
8.11.3	Named record type	67
8.11.4	Process declaration	67
8.11.5	Sequential declarations	68
9	The E-LOTOS base language	69
9.1	Introduction	69
9.2	Behaviours	69
9.2.1	Disabling behaviour expression	69
9.2.2	Synchronization behaviour expression	70
9.2.3	Concurrency behaviour expression	71
9.2.4	Selection behaviour expression	73
9.2.5	Suspend/Resume behaviour expression	73
9.2.6	Sequential composition	74
9.2.7	Action	75
9.2.8	Internal action	76
9.2.9	Successful termination without values	76
9.2.10	Successful termination	76
9.2.11	Inaction	77
9.2.12	Time block	77
9.2.13	Delay	77
9.2.14	Assignment	78
9.2.15	Nondeterministic Assignment	78
9.2.16	Choice over values	79
9.2.17	Trap	80
9.2.18	General parallel	81
9.2.19	Parallel over values	83
9.2.20	Variable declaration	84
9.2.21	Gate hiding	84
9.2.22	Renaming	85
9.2.23	Process instantiation	87
9.2.24	loop iteration	87
9.2.25	Case	88
9.2.26	Case with tuples	88
9.2.27	Signalling	89
9.3	Type expressions	90
9.3.1	Type identifier	90
9.3.2	Empty type	90
9.3.3	Universal type	90
9.3.4	Record type	90
9.4	Record type expressions	91
9.4.1	Singleton record	91
9.4.2	Universal record	91
9.4.3	Record disjoint union	91
9.4.4	Empty record	92
9.5	Value expressions	92
9.5.1	Primitive constants	92

9.5.2	Variables	92
9.5.3	Record values	92
9.5.4	Constructor application	93
9.6	Record value expressions	93
9.6.1	Singleton record	93
9.6.2	Record disjoint union	93
9.6.3	Empty record	93
9.7	Patterns	94
9.7.1	Record pattern	94
9.7.2	Wildcard	94
9.7.3	Variable binding	94
9.7.4	Expression pattern	95
9.7.5	Constructor application	95
9.7.6	Explicit typing	96
9.8	Record patterns	96
9.8.1	Singleton record pattern	97
9.8.2	Record wildcard	97
9.8.3	Record match	97
9.8.4	Record disjoint union	98
9.8.5	Empty record pattern	98
9.9	Record of variables	98
9.9.1	Singleton record variable	98
9.9.2	Record disjoint union	98
9.10	Behaviour pattern-matching	99
9.10.1	Single match	99
9.10.2	Multiple match	100
10	Predefined library	100
10.1	Booleans	101
10.2	Natural Numbers	103
10.3	Integral Numbers	106
10.4	Rational Numbers	109
10.5	Floating Point Numbers	112
10.6	Characters	113
10.7	Strings	114
10.8	Enumerated Type Scheme	114
10.9	Record Type Scheme	116
10.10	Set Type Scheme	117
10.11	List Type Scheme	120
A	Tutorial	122
A.1	The base language	122
A.1.1	Basic concepts	124
A.2	The module language	145
A.2.1	Basic concepts	148
A.3	An E-LOTOS specification of the ODP trader	156
A.3.1	Introduction	156
A.3.2	An overview of the ODP Trader	156
A.3.3	E-LOTOS Specification of the trader	158
A.3.4	The complete specification	161
B	Guidelines for LOTOS to E-LOTOS translation	179
B.1	Introduction	179
B.1.1	Specification and process definition	180
B.1.2	Basic LOTOS	180

B.1.3 Data Types 181

B.1.4 Full LOTOS 184

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 3.

In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75 % of the national bodies casting a vote.

Attention is drawn to the possibility that some of the elements of this International Standard may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

International Standard ISO/IEC 15437 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 7, *Software engineering*.

Annexes A and B of this International Standard are for information only.

Introduction

This International Standard contains the definition of the revised version of the LOTOS standard (ISO8807) known as E-LOTOS. The definition of E-LOTOS was based on the initial LOTOS goals enriched with feedback coming from the application of LOTOS to system design in industrial environments, as well as with the goal of supporting the ODP framework (Open Distributed Processing).

The semantics of the language is formed of a behavioural process algebra part which generalizes various LOTOS operators, and of a functional data definition part which is executable and more user friendly. The number of enhancements is high and all of them are very intertwined. A list is given now which tries to highlight from the user point of view the new features from which an E-LOTOS user should benefit when using the language. The most notable differences are:

- **Modularity:** E-LOTOS includes the following modularization facilities: module and interface definitions; export, import and visibility control; and generic modules. The modules can contain definitions of types, functions and/or processes.
- **Data typing.** E-LOTOS includes the following facilities for data definitions: predefined types, union types, recursive types, records, extensible records and record subtyping. The predefined data types and schemes are pervasive, i.e. they do not have to be declared before use in a module. As a consequence, they are directly part of the semantics model. At the semantic level, functions have been defined as a particular kind of processes which are deterministic and do not perform any visible event except termination and exception signaling.

A two level approach has been devised for backward compatibility with LOTOS. The first level is declarative. The second level provides executable definitions of the data types which should satisfy the declarative definition of level one. With this scheme, backwards compatibility with LOTOS is achieved by having declarative specifications at level one, either in ACT ONE or in an enhanced version of ACT ONE. Functional level two data definitions should satisfy level one specifications.

- **Time:** E-LOTOS models events as atomic and instantaneous occurrences. The specifier can define the time at which actions or behaviours may occur. For example, time restrictions can be added to action denotations restricting the occurrence to a given set of time instants or wait statements can be used to delay the occurrence of a behaviour in time.
- **Sequential composition:** a new operator `" ; "` is introduced which substitutes the operators existing in LOTOS for sequential composition, action prefix and exit/enabling (`>>`) pair. For example `a;B` or `B1;B2`. In LOTOS the `>>` operator always generates an internal action when the enabling is performed. In E-LOTOS, internal actions are generated only when necessary. This produces some minor differences with the semantics in LOTOS of some operators.
- **Unification of processes and functions at the semantic level.** A function is a process which performs only a termination action upon termination.
- **Introduction of write-many variables.** Write many variables are included in E-LOTOS, with a safe use assured by static semantic means. For example: assignment (write) must be performed before usage (read); no dangerous use of shared variables is made by parallel behaviours;
- **Introduction in E-LOTOS of output variables in processes and functions as the means to pass values in sequential composition.** This substitutes exit statements with value passing coupled with accept statements. The existence of input and output variables in processes and functions provides a unified approach for value communication among sequentially composed behaviours which is much more readable and concise. It is in line with notations used in ODP, like IDL (Interface Definition Language). To

illustrate the new approach to how sequential composition and value passing are performed let us see an example which includes variable assignment, processes and functions.

```
?x := 2;
phase1 [...] (x, ?result) ; (* a process *)
compute (result, ?res1, ?res2) ; (* a function *)
(lphase2 [...] (x, res1) ||| rphase2 [...] (x, res2))
```

- A more general parallel operator has been introduced in E-LOTOS which has the actual LOTOS parallel operator as a particular case. This operator is n-ary and supports the synchronization of J processes among K composed processes where $J \leq K$. Therefore synchronization patterns of 2 among N can be modeled in E-LOTOS. The new operator is also more readable because it clearly identifies the synchronizing gates for each behaviour composed.
- The Suspend/Resume operator which generalizes disabling. The new operator generalizes the LOTOS disabling operator by allowing a disabled behaviour to be resumed by a specific action of the disabling behaviour.
- Introduction of exceptions and exception handling in the behavioural and data parts with a uniform approach. Exception mechanisms permit new ways of structuring and are demanded by ODP.
- Explicit renaming operator for observable actions or exceptions. The renaming operator allows not only to change the name of the events occurring but to add and remove fields from the structure of events, or to merge and to split gates.
- Typed gates and partial synchronization. Gates must be explicitly typed in E-LOTOS. The use of the record subtyping relation permits partial synchronization of gates as well as provides backward compatibility with standard LOTOS untyped gates.

The introduction of enhancements has been performed trying to minimize the growth in the complexity of the language. Rather than introducing new operators, the enhancements have been introduced by generalizing existing operators. Only new statements and/or operators have been introduced when absolutely necessary. E-LOTOS is, as LOTOS, a language which permits a rich variety of specification styles which may be used to model different aspects of the design process. The styles existing in LOTOS, constraint oriented, resource oriented, state oriented, EFSM oriented, monolithic, ... can be also used in E-LOTOS. In addition the existence of exceptions, partial synchronization, event renaming, and other new constructs open new ways of specifying.

Information technology — Enhancements to LOTOS (E-LOTOS)

1 Scope

This International Standard defines the syntax and semantics of the enhanced LOTOS language (ISO 8807), named E-LOTOS. E-LOTOS is used for the formal description of the behavioural aspects of distributed and concurrent systems in general and in the area of open distributed processing in particular.

2 Conformance

A formal specification written in E-LOTOS conforms to the requirements of this International Standard if and only if it is derivable according to the syntactic rules defined in clause 5 and the semantics is unambiguously derivable from the semantic definition in clause 7.

3 Normative reference

The following normative document contains provisions which, through reference in this text, constitute provisions of this International Standard. For dated references, subsequent amendments to, or revisions of, any of these publications do not apply. However, parties to agreements based on this International Standard are encouraged to investigate the possibility of applying the most recent edition of the normative document indicated below. For undated references, the latest edition of the normative document referred to applies. Members of ISO and IEC maintain registers of currently valid International Standards.

ISO 8807, *Information processing systems — Open Systems Interconnection — LOTOS — A formal description technique based on the temporal ordering of observational behaviour*