INTERNATIONAL STANDARD

ISO 8807

First edition
1989-02-15

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

Systèmes de traitement de l'information — Interconnexion de systèmes ouverts — LOTOS — Technique de description formelle basée sur l'organisation temporelle de comportement observationnel

Reference number
ISO 8807 : 1989 (E)

Foreword

ISO (the International Organization for Standardization) is a worldwide federation of national standards bodies (ISO member bodies). The work of preparing International Standards is normally carried out through ISO technical committees. Each member body interested in a subject for which a technical committee has been established has the right to be represented on that committee. International organizations, governmental and non-governmental, in liaison with ISO, also take part in the work. ISO collaborates closely with the International Electrotechnical Commission (IEC) on all matters of electrotechnical standardization.

Draft International Standards adopted by the technical committees are circulated to the member bodies for approval before their acceptance as International Standards by the ISO Council. They are approved in accordance with ISO procedures requiring at least 75 % approval by the member bodies voting.

International Standard ISO 8807 was prepared by Technical Committee ISO/TC 97, *Information processing systems.*

Users should note that all International Standards undergo revision from time to time and that any reference made herein to any other International Standard implies its latest edition, unless otherwise stated.

Annex A forms an integral part of this International Standard. Annexes B, C, D and E are for information only.

Printed in Switzerland

CONTENTS

ANNEXES

TABLES

FIGURES

This page intentionally left blank

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

0 Introduction

0.1 General

Formal description techniques (FDTs) are methods of defining the behaviour of an (information processing) system in a language with a formal syntax and semantics, instead of a natural language such as English. In the following sub-clauses of this introduction, the importance of FDTs and their standardization is discussed. The objectives that an FDT must satisfy are considered. The origin of LOTOS is discussed. Finally the structure of this document is explained.

0.2 FDTs

Formal description techniques are important tools for the design, analysis and specification of information processing systems. It is by means of formal techniques that system descriptions can be produced that are *complete*, *consistent*, *concise*, *unambiguous* and *precise*. This is only possible if an FDT is self-contained, so that the descriptions given in an FDT need not refer to any informal knowledge of the system that is described. An important aspect of a formal system is that it allows analysis by mathematical methods. An FDT that has such a formal, mathematical basis can be used to prove the correctness of specifications.

0.3 The requirement for standard FDTs

If an FDT is defined in an International Standard, the description is available to all who require it. The Directives for the production of such a standard require a high degree of international acceptance and technical stability. Any amendment also requires international agreement. Hence a standard FDT offers the most useful form of presentation to those who wish to apply it.

0.4 The objectives to be satisfied by an FDT

Although this document describes an FDT that is generally applicable to distributed, concurrent information processing systems, it has been developed particularly for OSI. The main objectives to be satisfied by such an FDT are that it should be

a) *expressive*: an FDT should be able to define both the protocol specifications and the service definitions of the seven layers of OSI described in ISO 7498.
b) *well-defined*: an FDT should have a formal mathematical model that is suitable for the analysis of these specifications and definitions. The same model should support the checking of conformance of implementations that are permitted by the OSI International Standards. This model should also support the testing of an implementation for conformance.
c) *well-structured*: an FDT should offer means for structuring the description of a specification or definition in manner that is meaningful and intuitively pleasing. A good structure increases the readability, understandability, flexibility, and maintainability of system descriptions, and offers a better framework for their analysis.
d) *abstract*: there are two aspects of abstraction that an FDT should offer:
    1) an FDT should be completely independent of methods of implementation, so that the technique itself does not provide any undue constraints on implementors
    2) an FDT should offer the means of abstraction from irrelevant details with respect to the context at any point in a description. Abstraction can reduce the local complexity of system descriptions considerably. In the presence of a good structure, abstraction can help even further to reduce the complexity of descriptions.

0.5 The origin of LOTOS

LOTOS (Language of Temporal Ordering Specification) was developed by FDT experts from ISO/TC97 during the years 1981-1988. The basic idea that LOTOS developed from was that systems can be described by defining the temporal relation between events in the externally observable behaviour of a system. LOTOS has two components. The first component deals with the description of process behaviours and interactions, and is based on a modification of the Calculus of Communicating Systems (CCS), which was developed at the University of Edinburgh. The modification includes elements that were introduced in other calculi, which are related to CCS, viz. CSP and CIRCAL. Among the other theories that are related to CCS, and thus to LOTOS, are SCCS, MEIJE and ACP. CCS, and the related formal systems, provide a powerful analytical theory for concurrent processes.

The second component deals with the description of data structures and value expressions and is based on the abstract data type language ACT ONE. ACT ONE was developed at the Technical University of Berlin. The part of LOTOS dealing with the description of processes, i.e. dynamic behaviours, is not dependent upon ACT ONE. Many well-defined languages for the description of data structures could, in principle, be used in combination with the process definition facilities of LOTOS.

0.6 The structure of this International Standard

This document differs in contents from most International Standards. The importance of a formal, mathematical basis of an FDT (see clauses 0.2 and 0.4 b)) makes the inclusion of mathematical material in this definition necessary. Clause 4 introduces some fundamental mathematical concepts and notations that are used in the rest of the document. Clause 5 presents the fundamental mathematical structures that provide a semantic basis for LOTOS data types, behaviour expressions, and their combination. Clause 6 presents the syntax of the language and contains, together with 7.3, the rules for producing syntactically correct specifications; this part of the document requires knowledge of only basic mathematical concepts. Clause 7 presents the semantics of a LOTOS specification, based on the semantics of data types and behaviour expressions. Annex A contains the standard library of LOTOS data types and forms a part of this International Standard. The other annexes provide more information related to LOTOS, but do not form a part of the standard. In particular, annex C contains a *tutorial* on LOTOS, which is meant to provide a guide to the features of the language, and a convenient introduction to this standard for the non-technical reader.

1 Scope and field of application

This International Standard defines the syntax and semantics of the Formal Description Technique LOTOS. LOTOS is in general used for the formal description of distributed, concurrent information processing systems. In particular LOTOS can be used to describe formally the service definitions and protocol specifications of the layers of Open Systems Interconnection (OSI) architecture described in ISO 7498, and related standards, and conformance tests for implementations of OSI protocols and/or OSI functions. It can also be applied for the formal description of other distributed systems, such as telephone switching networks.

2 References

ISO 7498, *Information processing systems - Open Systems Interconnection - Basic Reference Model.*

CCITT Recommendation Z.100, *SDL*.

3 Conformance

A formal specification written in LOTOS conforms to the requirements of this International Standard if and only if it is derivable according to the syntactic rules defined in clause 6, and unambiguously defines a behaviour according to the semantics defined in clause 7.

This page intentionally left blank

4 Basic mathematical concepts and notation

This clause contains a list of basic mathematical concepts and related notations used in clauses 5,6, and 7.

4.1 General

$=_{df}$	is defined as.
iff	if and only if, i.e. double implication.

4.2 Sets

$\{a,b,c,...\}$	the *set* made up of elements $a,b,c,...$. The order in which the elements are listed is immaterial.
$\varnothing$	the *empty set*, i.e. the set having no elements.
$x \in A$	x is an *element* of the set A.
$x \notin A$	x is *not* an *element* of the set A.
$A \subseteq B$	A is a *subset* of B, i.e. all elements of A are also elements of B.
$A \cup B$	the *union* of A and B, i.e. the set which contains only all elements of A and all elements of B.
$\bigcup A$	the *union* of A, i.e. the set which contains only all elements of the elements of A (A must therefore be a set of sets).
$A \cap B$	the *intersection* of A and B, i.e. the set which contains only all elements of A which are also elements of B.
	sets A and B are *disjoint* iff $A \cap B = \varnothing$.
$A - B$	the *difference* of A and B, i.e. the set which contains only all elements of A which are not also elements of B.
$A \times B$	the *Cartesian product* of A with B, i.e. the set of all ordered pairs $<a,b>$, such that $a \in A$ and $b \in B$.
$A_1 \times A_2 \times ... \times A_n$	the *generalized Cartesian product* of $A_1,A_2,..,A_n$, i.e. the set of ordered n-tuples $<a_1,a_2,..,a_n>$ (see 4.3), such that $a_1 \in A_1$, $a_2 \in A_2$, .., $a_n \in A_n$.
$\{x \in A \mid Q(x)\}$	the set which contains only all those elements of A which satisfy property Q (the abbreviation $\{x \mid Q(x)\}$ is used where set A may be deduced from the context).

4.3 Lists

$a_1,..,a_n$, or $<a_1,..,a_n>$	the finite list (or sequence, or (n-)tuple) made up of the *elements*, or *components* $a_1,..,a_n$. Unlike sets, lists may contain more than one instance of the same element, since elements are distinguished by their position in the ordering of the list; an *ordered pair* is a list of two elements (e.g. $<a,b>$); the *empty* list has no elements and is denoted by $<>$. a *record* is an n-tuple of which each element is *labelled* with a unique label. If *lab* is the label of element x of record y then *y.lab* denotes x.
$<a_1,..,a_n> \cup <b_1,..,b_n>$	the componentwise union of lists, defined as: $<a_1 \cup b_1,..,a_n \cup b_n>$ ($a_1,..,a_n,b_1,..,b_n$ must therefore be sets or, again lists).
$\bigcup A$	The *union* (of the lists in) A, i.e. the componentwise union of only all elements of A (A must therefore be a set of n-tuples for a fixed n).
A^n	the set of all n-tuples with elements in A for fixed n.
A^*	is equivalent with $\bigcup\{A^n \mid n \in \mathbf{N}\}$, where **N** is the set of natural numbers.

4.4 Strings

$a_1 \ldots a_n$	the *string* made up of the elements $a_1, \ldots ,a_n$. A string is formed by the juxtaposition of its elements.
$s.t$	the *concatenation* of the strings s and t. The concatenation is the string that consists of the elements of s followed by those of t in the same order.
A^*	the set of all finite strings consisting of elements in the set A. This also includes the *empty string* ε, that has no elements.
$s \mid A$	the *restriction* of a string s to a set A is the string that consists of only all elements of s that are in A, in the order of their occurrence in s.

4.5 Relations and functions

$R \subseteq A \times B$	R is a *binary relation* between A and B, i.e. a set of elements of $A \times B$; the *domain* of R is defined as $\{a \in A \mid$ there exists some $b \in B$ such that $<a, b> \in R\}$; the *range* of R is defined as $\{b \in B \mid$ there exists some $a \in A$ such that $<a, b> \in R\}$;

if R is a binary relation between A and B, and S is a binary relation between B and C, their *composition* $R.S$ is the binary relation between A and C containing all and, only the pairs $<a,c>$ such that there exists some $b \in B$ with $<a,b> \in R$ and $<b,c> \in S$.

$f : A \rightarrow B$

f is a (*partial*) *function* from A to B, i.e. f is a binary relation between A and B such that for each $a \in A$, there exists at most one $b \in B$ such that $<a, b> \in f$;

if $<a,b> \in f$ then f is *defined* for a, and one may write $f(a) = b$;

the function f is *total* iff it is defined for all $a \in A$;

function composition is a special case of composition of binary relations, viz. when both relations are functions;

a function $f : A \rightarrow B$ is *injective* iff, for all a_1, a_2 in the domain of f, $f(a_1) = f(a_2)$ implies that $a_1 = a_2$;

the *inverse function*, $f^{-1} : B \rightarrow A$, of an injective function is defined as follows: for $b \in B$, if there exists an $a \in A$ such that $f(a) = b$, then $f^{-1}(b) = a$, otherwise f^{-1} is not defined; in symbols:
$f^{-1} : B \rightarrow A =_{df} \{<b, a> \mid f(a) = b\}$;

a function $f : A \rightarrow B$ is *'one-to-one'* iff it is injective and its inverse function is total.

4.6 Backus-Naur Form

The metalanguage used in this International Standard to specify the syntax of LOTOS is based on Backus-Naur Form (BNF). A BNF-like description of a language L is given by a set of *productions*, or *rewrite rules*. The meta-symbols used to compose rewrite rules are listed in table 1.

A *terminal* (*symbol*) is a symbol that appears literally in L. A *nonterminal* (*symbol*) is a symbol that denotes a syntax construct of L (which is ultimately represented by a string of terminal symbols).

A rewrite rule has the format:

nonterminal-symbol = meta-expression .

where the meta-expression is an expression constructed using terminal and nonterminal symbols, and the operators listed in table 1. A meta-expression contains at least one terminal or non-terminal symbol. Adjacent terminal and/or nonterminal symbols occurring in a meta-expression denote the concatenation of the text they ultimately represent. The nature of this concatenation is subject to the rules given in 6.1.

A meta-expression denotes, depending on the operators used in it, a set S of terminal/nonterminal sequences. A rewrite rule is interpreted as follows: the nonterminal symbol at the left-hand side can be replaced by any one of the sequences of S.

All operators (implicit concatenation included) have precedence over the alternative operator.