INTERNATIONAL STANDARD

ISO 13209-2

First edition
2012-08-15

Road vehicles — Open Test sequence eXchange format (OTX) —

Part 2: Core data model specification and requirements

Véhicules routiers — Format public d'échange de séquence-tests (OTX) —

Partie 2: Exigences et spécifications du modèle de données central

Reference number
ISO 13209-2:2012(E)

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.org
Web www.iso.org

Published in Switzerland

Contents

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.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

The main task of technical committees is to prepare International Standards. Draft International Standards adopted by the technical committees are circulated to the member bodies for voting. Publication as an International Standard requires approval by at least 75 % of the member bodies casting a vote.

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

ISO 13209-2 was prepared by Technical Committee ISO/TC 22, *Road vehicles*, Subcommittee SC 3, *Electrical and electronic equipment*.

ISO 13209 consists of the following parts, under the general title *Road vehicles — Open Test sequence eXchange format (OTX)*:

- *Part 1: General information and use cases*
- *Part 2: Core data model specification and requirements*
- *Part 3: Standard extensions and requirements*

Introduction

Diagnostic test sequences are utilized whenever automotive components or functions with diagnostic abilities are being diagnosed, tested, reprogrammed or initialised by off-board test equipment. Test sequences define the succession of interactions between the user (i.e. workshop or assembly line staff), the diagnostic application (the test equipment) and the vehicle communication interface as well as any calculations and decisions that have to be carried out. Test sequences provide a means to define interactive, guided diagnostics or similar test logic.

Today, the automotive industry mainly relies on paper documentation and/or proprietary authoring environmments to document and to implement such test sequences for a specific test application. An author who is setting up engineering, assembly line or service diagnostic test applications needs to implement the required test sequences manually, supported by non-uniform test sequence documentation, most likely using different authoring applications and formats for each specific test application. This redundant effort can be greatly reduced if processes and tools support the OTX concept.

ISO 13209 proposes an open and standardized format for the human- and machine-readable description of diagnostic test sequences. The format supports the requirements of transferring diagnostic test sequence logic uniformly between electronic system suppliers, vehicle manufacturers and service dealerships/repair shops.

This part of ISO 13209 represents the requirements and technical specification for the fundament of the OTX format, namely the "OTX Core". The Core describes the basic structure underlying every OTX document. This comprises detailed data model definitions of all required control structures by which test sequence logic is described, but also definitions of the outer, enveloping document structure in which test sequence logic is embedded. To achieve extensibility the core also contains well-defined extension points that allow a separate definition of additional OTX features – without the need to change the core data model.

ISO 13209-3 extends the Core by a set of additional features, using of the Core extension mechanism (which may also be applied for proprietary extensions).

This part of ISO 13209 is the most generic and stand-alone part of ISO 13209. In principle, it is also applicable in other areas for any sequential logic description, even outside the automotive domain. Automotive-specific features are therefore contained solely in ISO 13209-3.

Road vehicles — Open Test sequence eXchange format (OTX) —

Part 2:
Core data model specification and requirements

1 Scope

This part of ISO 13209 defines the OTX Core requirements and data model specifications.

The requirements are derived from the use cases described in ISO 13209-1. They are listed in the requirements section which composes the first major part of this document.

The data model specification aims at an exhaustive definition of all OTX Core features implemented to satisfy the Core requirements. Since OTX is designed for describing test sequences, which themselves represent a kind of program, the Core data model follows the basic concepts common to most programming languages.

Thus, this part of ISO 13209 establishes rules for syntactical entities like parameterised procedures, constant and variable declarations, data types, basic arithmetic, logic and string operations, flow control statements like loop, branch or return, simple statements like assignment or procedure call as well as exception handling mechanisms. Each of these syntactical entities is accompanied by semantic rules which determine how OTX documents are to be interpreted. The syntax rules are provided by UML class diagrams and XML schemas, whereas the semantics are given by UML activity diagrams and prose definitions.

With respect to documentation use cases, special attention is paid to defining a specification/realisation concept (which allows for "hybrid" test sequences: human readable test sequences that are at the same time machine-readable) and so called floating comments (which can refer to more than one node of the sequence).

The Core data model does NOT define any statements, expressions or data types that are dependent on a specific area of application.

2 Normative references

The following referenced documents are indispensable for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO/IEC 10646:2011, *Information technology — Universal Coded Character Set (UCS)*

ISO 13209-1, *Road vehicles — Open Test sequence eXchange format (OTX) — Part 1: General information and use cases*

ISO/IEC 19501:2005, *Information technology — Open Distributed Processing — Unified Modeling Language (UML) Version 1.4.2*

ISO 22901 (all parts), *Road vehicles — Open diagnostic data exchange (ODX)*

IEEE 754:2008, *IEEE Standard for Floating-Point Arithmetic*

RFC 2045, *Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies*

RFC 2046, *Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types*

RFC 4122, *A Universally Unique IDentifier (UUID) URN Namespace*

W3C XSD:2004, *W3C Recommendation: XML Schema (all parts)*

W3C XML:2008*, W3C Extensible Markup Language (XML) 1.0 (Fifth Edition)*

W3C XMLNS:2009*, W3C Recommendation: Namespaces in XML 1.0 (Third Edition)*

W3C XMLBASE:2009*, W3C Recommendation: XML Base (Second Edition)*

W3C XLink:2010*, W3C Recommendation: XML Linking Language (XLink) Version 1.1*

3 Terms, definitions and abbreviated terms

3.1 Terms and definitions

For the purposes of this document, the terms and definitions given in ISO 13209-1 and the following apply.

3.1.1
attribute
<UML> a property of a UML class

3.1.2
attribute
<XSD/XML> named property of an XSD complex type or an XML element

3.1.3
after market
part of the automotive industry concerned with manufacturing, remanufacturing, distribution, retailing, and installation of all vehicle parts, chemicals, tools, equipment and accessories for light and heavy vehicles, after the sale of the automobile by the original equipment manufacturer (OEM) to the consumer

3.1.4
after sales
after sales department
department of an automotive OEM that is concerned with the distribution, retailing, servicing, repair and installation of vehicles of that OEM

3.1.5
constant
identifier of a non-writable memory location

3.1.6
context
environmental circumstances which influence test sequence execution

NOTE OTX test sequences can be configured to behave differently according to different context situations. Contextual information depends on factors such as the particular vehicle that is currently attached to the test application (e.g. the current vehicle's model type, the engine type, etc.), on the test application settings (e.g. a setting controlling whether the test sequence shall run in debug mode) or on other factors such as whether the test sequence is running in a manufacturing or a service workshop environment, etc.

3.1.7
engineering
engineering department
department of an automotive OEM which is concerned with the design, development, integration and testing of vehicles of that OEM

3.1.8
expression
syntactical construct which describes a specific computation with a set of arguments and a single return value

3.1.9
identification routine
method or software by which a diagnostic application identifies contextual information

3.1.10
manufacturing
manufacturing department
department of an automotive OEM which is concerned with the production and end-of-line testing of vehicles of that OEM

3.1.11
original equipment manufacturer
OEM
automotive company that engineers, manufactures, sells and services vehicles

3.1.12
OTX Core
most generic and stand-alone part of the overall OTX data model which describes the basic structure underlying every OTX document and comprises detailed data model definitions of all required control structures (loops, branches, …) by which test sequence logic is described, but also definitions of the outer, enveloping document structure in which test sequence logic is embedded

3.1.13
OTX Extension
OTX Standard Interface Definition
otxIFD
set of OTX data type-, action-, term- and signature-definitions that are tailored for a specific area of application and that are defined aside of the OTX Core

NOTE OTX Extensions model the data types, actions, terms and signatures needed for communication through diverse interfaces. By using these interfaces, calls can be performed to external systems whose internal behaviour does not have to be known to the (client) OTX test sequence/runtime. The system-side interface (server-side) can be proprietary because the adapter design pattern is applied.

3.1.14
procedure signature
description of the interface of an OTX procedure

3.1.15
reference
value which refers to data in memory

3.1.16
session
instance of test sequence execution

3.1.17
term
value described by and computed from an expression

3.1.18
test sequence
test procedure defining a full test

NOTE A test sequence is a procedure also, but not all procedures are test sequences. In an OTX document, the procedure representing a test sequence shall be named "main". By using procedures, a test sequence may be split into several procedure modules. An adequately assembled set of frequently needed procedures may serve as a library which provides procedures that can be called from any other (client) procedure or test sequence.

3.1.19
test procedure
procedure
stand-alone, parameterisable flow of OTX actions that can be called from other OTX procedures

3.1.20
validity
Boolean context variable, global Boolean constant or a named Boolean expression used for activating/deactivating parts of the OTX test sequences according to the current context situation

NOTE Parts of OTX test sequences which are marked with a validity name shall be executed only if the associated Boolean expression is true according to the current context situation.

3.1.21
variable
identifier of a writable memory location

NOTE The term "variable" is used as a collective term for document scope variables, local variables, non-constant parameters and also items in non-constant lists or maps or other compound data structures. In OTX, these can be addressed by giving the identifier of the variable or parameter, optionally accompanied by a path into compound data structures which allows the inner parts of variables or parameters to be addressed.

3.2 Abbreviated terms

API	Application Programming Interface
IFD	Interface Definition (OTX extension)
JRE	Java Runtime Environment
NOP	No Operation Performed
OEM	Original Equipment Manufacturer
OTX	Open Test sequence eXchange
UML	Unified Modeling Language
XML	Extensible Markup Language
XSD	XML Schema Definition

4 Requirements

4.1 General

Since OTX is merely a static data format and not a software application, it has to be kept in mind that all following requirements are related to static format features and **not** to the behaviour of any OTX based software product. As a matter of course, all such products are indirectly affected by the requirements given here in so far that they shall be able to write, read or execute valid OTX documents according to the rules given in this document. Aside from that, requirements towards any such product are not in the scope of this specification.

4.2 Basic principles for requirements definition

Basic principles have been established as a guideline to define the OTX requirements:

a) OTX requirements specify the conditions that the OTX data model and format shall satisfy.

b) All stakeholders (System Suppliers, OEMs, Tool Suppliers), which offer diagnostic test procedures are expected to implement and follow the requirements of this part of ISO 13209.

c) The content of OTX documents and the quality of the information is the responsibility of the originator.

4.3 Clustering of requirements

Table 1 provides an overview of the main categories of OTX requirements. Each category may have one or more requirements.

Table 1 — Main requirements clustering

#	Main title of requirement cluster	Brief description
1	general format and language requirements	Requirements regarding the general aspects like the chosen programming paradigm, file format (XML), …
2	test sequence development process support	Requirements about different stages in the test procedure authoring process, outlining human-readable (documentation) vs. machine-readable (execution) test procedures.
3	language feature details	Requirements concerning details like declarations, data types, expressions, statements, etc.
4	boundaries	Features that should NOT be part of OTX

4.4 Requirement priorities

Each of the following requirements carries a priority-attribute which can be set to SHALL or SHOULD.

- SHALL:
  The requirement represents stakeholder-defined characteristics the absence of which will result in a deficiency that cannot be compensated by other means.

- SHOULD:
  If the requirement defined characteristic is not or not fully implemented in the data model, it does not result in a deficiency, because other features in the data model can be used to circumvent this.

4.5 General format and language aspects

Core_R01 – Machine readable format

Priority: SHALL

Rationale: The focus of OTX is on the exchange of data between tools in the vehicle diagnostic process. To leverage highest efficiency, the tools shall be able to operate automatically on OTX files (e.g. for importing and exporting of OTX-relevant data)

Description: The OTX format has to be machine-readable to allow a tool to open an existing document for editing, checking, displaying or executing.

Core_R02 – Platform independence

Priority: SHALL

Rationale: If OTX would bind to specific Hardware, Operating System or Application, its potential usages are diminished and applicability of the standard is decreased.

Description: OTX shall not be dependent on any specific hardware or software platform. OTX shall not be bound to any particular hardware, operating system or application.

Core_R03 – Well-defined syntax and semantics

Priority: SHALL

Rationale: OTX shall be a machine-readable data format. This implies an unambiguously defined syntax and semantics.

Description: All OTX elements have to be defined clearly (syntax + semantics). For the syntax definition, XML Schema shall be used. For the behavioural/semantics specification, a prose description shall exist.

Core_R04 – Universal language

Priority: SHALL

Rationale: Diagnostic applications can be seen as domain specific computer programs. These require complex computations and no limits are known or foreseen today that allow OTX to be restricted with respect to Turing-completeness.

Description: OTX shall have the ability to solve any computable problem. (Turing-Completeness)

NOTE 1 Legacy sequences can (theoretically) be transformed to OTX and back, if the legacy sequence format and OTX are Turing-complete.

Core_R05 – Minimal language

Priority: SHOULD

Rationale: Fulfilment of this requirement reduces the implementation effort necessary to integrate OTX into tools and is thus a very relevant market-driving factor for OTX.

Description: OTX should be defined with the minimal set of language elements necessary to reach Turing-Completeness.

NOTE 2 OTX should not be designed for comfort of expressing computational programs (as are programming languages like Java, C++ or Delphi), but rather for effectiveness of transporting diagnostic application knowledge unambiguously between different tools/parties in the diagnostic process.

Core_R06 – Structured programming approach

Priority: SHALL

Rationale: Structured programming can be seen as a subset or sub discipline of procedural programming, one of the major programming paradigms. It removes reliance on the GOTO statement for controlling the flow of a program. Using GOTO statements in programming often leads to a complex, tangled and unreadable control structure, which is clearly not desired in OTX.

Description: OTX shall follow the structured programming approach. Only flow control statements branch, loop return, continue, break and throw may implicitly induce jumps. The behaviour of these jumps shall be well defined in the prose semantic documentation of each of these statements. An explicit GOTO statement which allows to jump anywhere in the procedure **shall not** be supported

Core_R07 – Imperative structure

Priority: SHALL

Rationale: Test procedures are usually considered as a procedure of commands that need to be executed one after one by a runtime system. Since the imperative programming paradigm matches exactly for this concept, it is well suited for OTX.

Description: OTX shall only support program structures that can be translated by a compiler into imperative programming languages.

Core_R08 – Extensibility

Priority: SHALL

Rationale: The scope of diagnostic applications in the diagnostic process is wide. Engineering, Production and After Sales applications interface numerous and diverse devices, server applications and modules, which cannot be completely addressed with the first release of the standard and which evolve over time.

Description: OTX shall be extendable to integrate means to access new technology employed within the diagnostic process. It shall be possible to integrate interfaces of various base technologies into OTX.

4.6 Test sequence development process support

Core_R09 – Embed non-machine readable content

Priority: SHALL

Rationale: Use cases will occur where diagnostic applications shall be expressed in OTX but e.g. the interfaces to all used devices are not available (e.g. how to communicate to a nut runner). In this case it would be preferable to express the diagnostic application in OTX and express the non-standardized device access in prose or in pseudo code. An OTX-compliant tool could then import such a file and mark the parts of the diagnostic application that need to be replaced with executable content by a diagnostics engineer.

Description: OTX shall provide means to express parts of a diagnostic application in a non-machine readable format. This non-machine readable content shall be clearly marked so that processes operating on OTX files can identify it.

Core_R10 – High level test procedure

Priority: SHALL

Rationale: In a step-wise test procedure design process, it might become necessary to specify procedures in prose-form only. Skeletal control structures might already be part if this high-level description, but the details of implementation might not be known at design time (loop conditions, exact service names …).

Description: It shall be possible to describe test procedures at a high level.

Core_R11 – Exchange high level test procedure

Priority: SHALL

Rationale: A test procedure specified in prose-form only shall nevertheless pose a valid OTX document, even though it is not executable.

Description: It shall be possible to exchange a high-level test plan using a plain text description.

Core_R12 – Exchange a fully functional test procedure

Priority: SHALL

Rationale: A test procedure containing no prose-form, but only implementation details shall nevertheless pose a valid OTX document, even though it is not easily human-readable.

Description: It shall be possible to mix high-level description and implementation details on the same procedure.

Core_R13 – Exchange an intermediate stage test procedure

Priority: SHALL

Rationale: A test procedure containing a mix of prose and fully implemented parts shall nevertheless pose a valid OTX document.

Description: It shall be possible to mix high-level description and implementation details on the same procedure.

Core_R14 – Floating comments

Priority: SHALL

Rationale: Situations will occur where comments are needed that can be freely attached to parts of the flow of commands in a test procedure. Such comments shall not be locally bound or contained within single statements; they shall be defined aside from the flow and only point to parts of it. Comments are purely informational nodes that shall not be relevant for execution of a test procedure.

Description: It shall be possible add floating comments to a test procedure that can refer to one or more statements its flow or its sub-flows at any block depth.

4.7 Language feature details

4.7.1 Declarations

Core_R15 – Declarations

Priority: SHALL

Description: OTX has to support the declaration of constants and variables as well as test procedure parameters. A declaration shall contain a name, a data type, an optional initialisation value and an optional description.

Core_R16 – Initialisation

Priority: SHALL

Rationale: It shall be possible to set the initial value for an identifier to a value other than the default.

Description: OTX shall support the optional initialisation of declared identifiers.

Core_R17 – Constant declarations

Priority: SHALL

Rationale: There will be cases when an OTX author wants to guarantee that the value of an identifier in the test procedure can not be changed. Therefore the author needs to have a means to mark an identifier as a constant. The value of a constant is not allowed to change during the lifetime of the constant.

Description: OTX shall support the declaration of constants. Constants shall be set with the declaration (initialisation is mandatory).

Core_R18 – Variable declaration

Priority: SHALL

Rationale: In order to reflect the fact that an identifier can change its value during procedure execution, a means for marking identifiers as variables is needed. The value of a variable is allowed to change during procedure execution; new values can be assigned to it.

Description: OTX shall support the declaration of variables.

Core_R19 – Input parameter declarations

Priority: SHALL

Rationale: When a test procedure is called, it will occur that information needs to be passed to the called procedure.

Description: OTX shall support declaration of any number of input parameters that are passed to the test procedure from the caller.